RL-TR-93-117, Vol I (of two)
Final Technical Report
June 1993

CADBIT II - COMPUTER-AIDED DESIGN FOR BUILT-IN TEST

Hughes Missile Systems Company

Michael Davis, Sonny Kwan, Tony Holzer

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

Rome Laboratory
Air Force Materiel Command
Griffiss Air Force Base, New York

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RL-TR-93-117, Vol I (of two) has been reviewed and is approved for publication.

APPROVED:

EDWARD L. DEPALMA
Project Engineer

FOR THE COMMANDER:

ANTHONY J. FEDUCCIA
Acting Director, Reliability
Electromagnetics and Reliability Directorate

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify RL (ERSD) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1993	Final - - -

4. TITLE AND SUBTITLE
CADBIT II - COMPUTER-AIDED DESIGN FOR BUILT-IN TEST

5. FUNDING NUMBERS
C - F30602-90-C-0116
PE - 62702F
PR - 2339
TA - 02
WU - 4U

6. AUTHOR(S)
Michael Davis, Sonny Kwan, Tony Holzer

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Hughes Missile Systems Company
P.O. Box 85357
San Diego CA 92186

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Rome Laboratory (ERSD)
525 Brooks Road
Griffiss AFB NY 13441-4505

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
RL-TR-93-117, Vol I (of two)

11. SUPPLEMENTARY NOTES
Rome Laboratory Project Engineer: Edward L. DePalma/ERSD/(315)330-2702.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The objective of CADBIT II was to develop a CAD workstation-based system with software to help designers incorporate Built-In Test (BIT) in their CAD-based circuit board designs. The CADBIT II system consists of a Unix-based workstation, electronic CAD software, and the contractor-developed CADBIT II software module. CADBIT II benefits printed circuit board designers through four major functions. The Tutorial function increases awareness and understanding of Built-In Test techniques. This function offers on-line access to an extensive database of information on current techniques used to implement built-in test in digital, analog, and hybrid (mixed digital and analog) circuit boards. The Selection function helps designers choose the best technique for their design. The selection log compares characteristics and attributes of each BIT technique in the CADBIT database to information about the user's specific "Circuit Under Test". The CAD Insertion function helps designers implement a selected BIT technique in a CAD schematic. This function provides a "Default Design: with a consistent set of components to implement the selected BIT technique. The Evaluation function analyzes the design impact of adding Built-In-Test circuitry based on total board area, weight, and power "penalties" for each BIT.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Computer Aided Design, Built-In-Test, BIT, BILBO, Comparator, Software Design Tool, Mentor Graphics, Testability	88
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	U/L

TABLE OF CONTENTS

TABLE OF CONTENTS, Continued

LIST OF FIGURES

LIST OF FIGURES, Continued

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

ADC	Analog-to-Digital Converter
ALS	Advanced Low-Power Schottky
AMPLE	Advanced Multi-Purpose Language
ATLAS	Advanced Tactical Ladar System
BIT	Built-In Test
BTID	BIT Technique Insertion Diagram
CAD	Computer-Aided Design
CADBIT	Computer-Aided Design for Built-In Test
CDE	Component Determination Equation
CLIN	Contract Line Item Number
CPU	Central Processing Unit
CUI	Common User Interface
CUT	Circuit Under Test
DAC	Digital-to-Analog Converter
DARPA	Defense Advanced Research Project Administration
ECC	Error Correction Code
EDCC	Error Detection and Correction Code
FAT	Final Acceptance Test
GMR	Good Machine Response
HMSC	Hughes Missile Systems Company
H/W	Hardware
IPC	Interprocess Communication
IR	Infrared
OSF	Open Systems Foundation
PAT	Preliminary Acceptance Test
PCB	Printed Circuit Board
PE	Penalty Equation
RAM	Random Access Memory
RL	Rome Laboratory
ROM	Read-Only Memory

LIST OF ACRONYMS AND ABBREVIATIONS, Continued

SCCS	Source Code Control System
SDD	Software Design Document
SDP	Software Development Plan
SP	System Processor
SPCR	Software Problem/Change Report
StP	Software Through Pictures
STP	Software Test Plan
SUM	Software User's Manual
S/W	Software
TIM	Technical Interchange Meeting
URD	User Requested Data
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
VLSI	Very Large Scale Integration

SUMMARY

This report describes the results of the CADBIT II (Computer-Aided Design for Built-In Test II) program sponsored by Rome Laboratory and performed by Hughes Missile Systems Company during the period from September 1990 to November 1992. The objective of CADBIT II was to develop a CAD workstation-based system with software to help designers incorporate Built-In Test (BIT) in their CAD-based circuit board designs. The CADBIT II system consists of a Unix-based workstation, electronic CAD software, and the contractor-developed CADBIT II software module.

CADBIT II benefits Printed Circuit Board designers through four major functions:

The Tutorial function increases awareness and understanding of Built-In Test techniques. This function offers on-line access to an extensive database of information on current techniques used to implement built-in test in digital, analog, and hybrid (mixed digital and analog) circuit boards. The tutorial information consists of text and graphics including block diagrams and detailed schematics illustrating connection of built-in test circuitry.

The Selection function helps designers choose the best technique for their design. The selection logic compares characteristics and attributes of each BIT technique in the CADBIT database to information about the user's specific "Circuit Under Test." CADBIT determines which BIT techniques are suitable for the user's circuit design, and then ranks the suitable BIT techniques to highlight those with the least impact on board area, weight, power, and timing constraints.

The CAD Insertion function helps designers implement a selected BIT technique in a CAD schematic. This function provides a "Default Design" with a consistent set of components to implement the selected BIT technique. The Insertion function "scales" the default design, calculating the appropriate part quantities for the user's circuit. CAD symbols for the correct number of parts are then inserted into the user's schematic and step-by-step guidance is offered for connection of the inserted BIT circuitry.

The Evaluation function analyzes the design impact of adding built-in test circuitry. The Evaluation function calculates the total board area, weight, and power "penalties" incurred by addition of BIT circuitry.

A major feature of CADBIT II is its "BIT Library" providing designers on-line access to information about current BIT techniques. The BIT Library can be modified to meet the user's present and future needs. Modular design and utility software enable users to maintain and expand the BIT techniques in the Library as the state of the art advances. The CADBIT II software and BIT Library were based on requirements and data provided from the CADBIT I program which ended in September 1988.

Computer standards, including Unix, C, and the X-Windows-based OSF/Motif graphical user interface, were employed to maximize portability, maintainability, and user familiarity and acceptance. CADBIT II software was integrated with an industry-leading electronic Computer-Aided Design (CAD) software package.

CADBIT II helps designers improve self-testability of their printed circuit boards. The software provides an excellent building block for future development aimed at integrating reliability, maintainability, and testability in a Computer-Aided Design environment.

1.0 INTRODUCTION

What is CADBIT? As Figure 1.0 illustrates, the definition of CADBIT has several important elements. CADBIT is an integrated system which includes commercial CAD workstation hardware and electronic CAD software. The commercial hardware and software is integrated with a contractor-developed software module which helps designers learn about available BIT techniques, select the most suitable techniques for their designs, insert BIT circuitry into their CAD schematics, and evaluate the design impact of added BIT. CADBIT is intended for use by designers concerned with improving self-testability of their printed circuit boards.

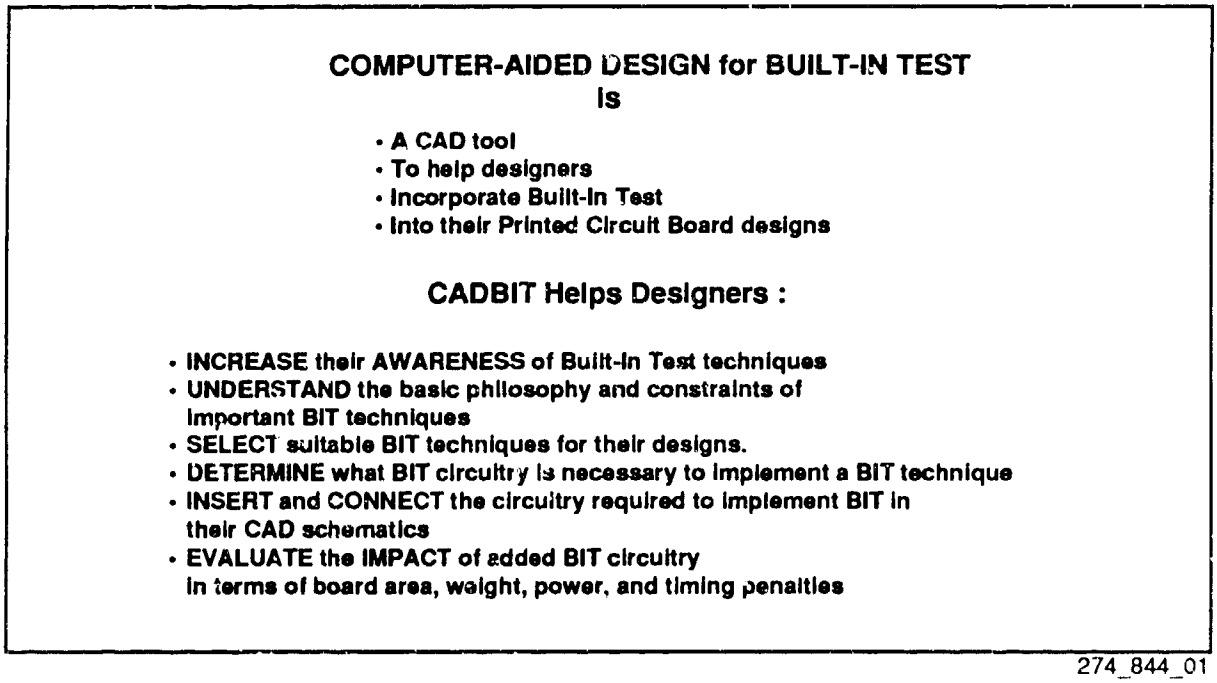


Figure 1.0 What is CADBIT?

CADBIT II is a continuation of the CADBIT I (or CAD-BIT) program funded by Rome Laboratory and performed by Grumman Aerospace during the period from September 1986 to September 1988. The purpose of the CADBIT I program was to develop design requirements, the associated Built-In Test (BIT) database, and a software specification for the CADBIT system. The goal was to use open software standards so that the CADBIT module would be capable of operating on generic Unix CAD workstations.

The CADBIT II program took the concepts developed in CADBIT I and moved them through the software and database design, development, and test phases. The CADBIT II program objective, as illustrated in Figure 1.1, was to develop a software module that would provide the design engineer the capability to integrate BIT into the design of Printed Circuit Boards (PCBs).

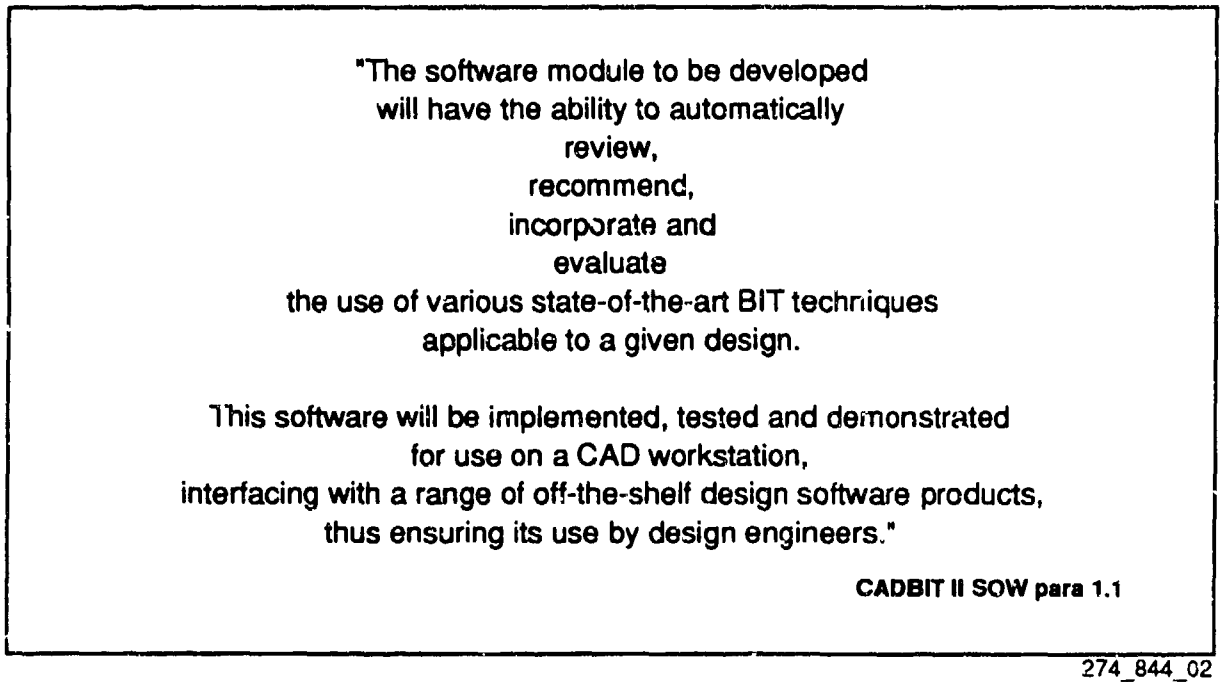


Figure 1.1 CADBIT II Objective

1.1 SCOPE

CADBIT II developed a CAD workstation system based on requirements and BIT technique data provided in the CADBIT I Final Report, as indicated in Figure 1.2.

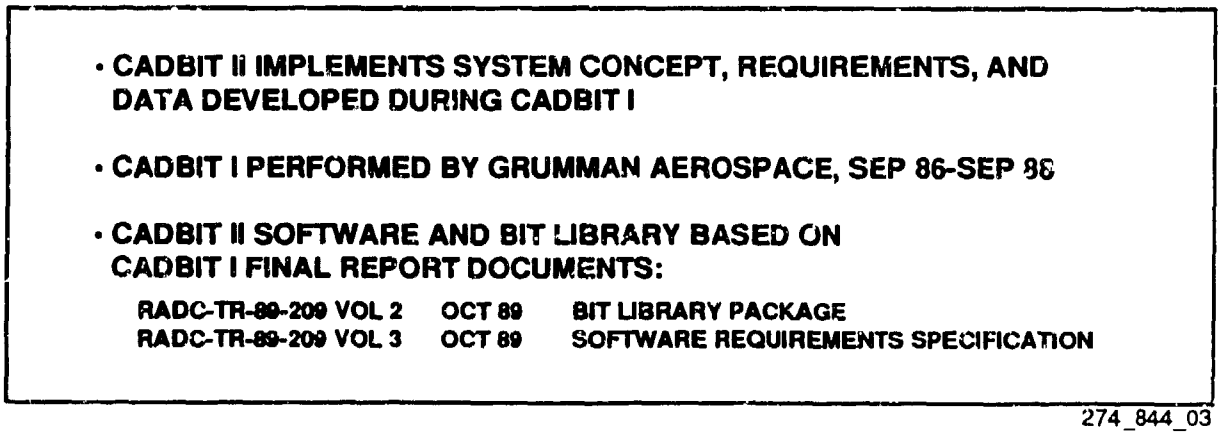


Figure 1.2 CADBIT II software and BIT Library are based on CADBIT I requirements.

CADBIT II focuses on BIT techniques suitable for printed circuit boards. Future development could expand the system to include both system- and chip- level techniques as well.

CADBIT II BIT technique Insertion and Evaluation functions are performed on "graphic-" rather than "text-based" circuit models. That is, CADBIT II assumes the designer is entering the circuit design in the graphic format of a CAD Schematic Capture software package. Future versions of CADBIT may expand these functions to help designers "insert" BIT by modifying text-based designs -- for example, circuit designs captured in a Hardware Description Language such as VHDL.

To increase portability, CADBIT II software was developed using de facto computer standards including Unix, C, and the X Windows-based OSF/Motif graphical user interface. CADBIT II developed a workstation-based system integrated with an industry-leading electronic CAD software package. Porting the CADBIT software to other computer platforms, especially low-cost personal computers, and integration with other commercial CAD software packages would be an excellent growth path to increase accessibility and flexibility.

1.2 FINAL REPORT ORGANIZATION

The CADBIT II Final Report is organized into two volumes. Volume I contains an overview of the CADBIT II system and the system development effort. Volume II contains details of the Built-In Test (BIT) technique data in the CADBIT II "BIT Library."

This first volume of the Final Report is organized into sections as follows:

Section 1 provides an introduction to the CADBIT II effort and the resulting CADBIT II system. This section includes an overview of the system, discusses the scope of the CADBIT II effort, and the organization of the final report.

Section 2 outlines the basic elements of the contract including deliverables, schedule, and major tasks and milestones.

Section 3 addresses the elements of the CADBIT II system - the CAD workstation, the Printed Circuit Board (PCB) design software, and the contractor-developed software module.

Section 4 introduces the "BIT Library" used by CADBIT II. The details of the BIT Library are presented in Volume II.

Section 5 discusses CADBIT II system functions and operation, including the four basic designer-oriented functions of BIT Tutorial, Selection, Insertion, and Evaluation.

Sections 6 and 7 cover CADBIT II software and system development and test.

Section 8 discusses the two Printed Circuit Board testbeds used to test the CADBIT II system and the VHDL model used to test the system's ability to accept VHDL-formatted files.

Section 9 presents conclusions and recommendations stemming from the CADBIT II effort.

Section 10 provides a list of documents referenced in the Final Report.

2.0 CONTRACT OVERVIEW

The CADBIT II effort was performed during the period from September 90 to November 92. This section provides an overview of the major tasks, contract deliverables, program schedule, major milestones and program reviews.

2.1 MAJOR TASKS

Figure 2.0 shows the major contract tasks. CAD workstation hardware and software are discussed in Section 3. The BIT Library, containing BIT technique data required for CADBIT II, is discussed in Section 4 and the entire second volume of this Final Report. Software development is discussed in Section 6 and software test is covered in Section 7. Application of CADBIT II to testbed printed circuit boards is discussed in Section 8. Software delivery, installation, and demonstration are covered in Section 2.3.

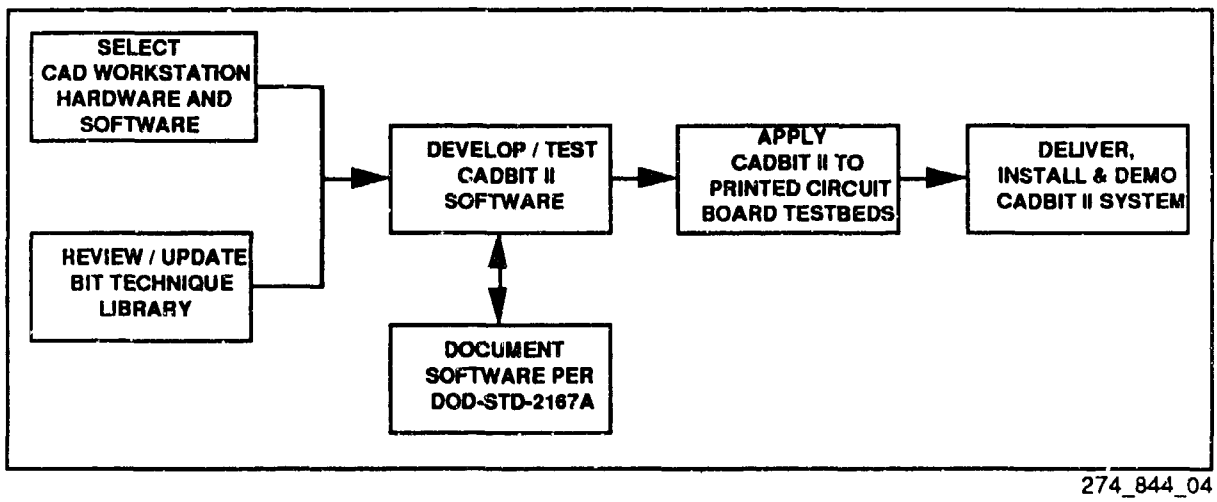


Figure 2.0 CADBIT II Major Tasks

2.2 DELIVERABLES

CADBIT II contract deliverables were organized into two Contract Line Items.

Contract Line Item Number 1 (CLIN 1), titled "Design/Develop (Hardware / Software) System" consisted of the workstation hardware and software, the Printed Circuit Board design software, and the contractor-developed CADBIT II software.

CLIN 2 ("Data") consisted of 7 Contract Data Requirements List (CDRL) items shown in Figure 2.1. The CDRL items included four software documentation items in accordance with DOD-STD-2167A, manuals for commercial hardware and software, monthly status reports, and this final report.

CDRL# (or CONTRACT REF)	DELIVERABLE ITEM	DUE	COMPLETE
CLIN 1	DESIGN/DEVELOP (SOFTWARE/HARDWARE) SYSTEM	24.75 MAC	10/23/92
Attch 2 Seq 1	CADBIT Software	CLIN 1	10/16/92*
SOW 4.1.2.5	CADBIT Workstation (H/W and S/W)	CLIN 1	10/16/92*
CLIN 2	DATA	w/Final Rpt	
A001	Monthly Status Reports	Monthly	11/22/92
A002	Software Development Plan (SDP)	CLIN 1	10/16/92
A003	Software Design Document (SDD)	CLIN 1	10/16/92
A004	Software Test Plan (STP)	FAT	10/12/92
A005	Commercial-Off-The-Shelf (COTS) Manuals	4 MAC	09/22/92
A006	Software User's Manual (SUM)	CLIN 1	10/16/92
A007	Final Report - Draft	CLIN 1+30	11/22/92*

CDRL = Contract Data Requirements List
CLIN = Contract Line Item Number
FAT = Final Acceptance Test
SOW = Statement of Work

*Notes:
-Final software tape delivered with Final Report draft
-Second workstation delivered with Final Report draft
-Corrected Final Report due CLIN 1 + 130 = 3/2/93

274_844_10

Figure 2.1 CADBIT II Contract Deliverables

2.3 DELIVERIES

WORKSTATION DELIVERIES

Two HP/Apollo workstations were delivered under the CADBIT II contract. The configuration of these two workstations is shown in Table 2.0. Workstation 1 was delivered to Rome Laboratory in the spring of 1991 and was upgraded to Mentor Graphics Version 8 in the summer/fall of 1992. Workstation 2 was used by HMSC in San Diego to develop the CADBIT II System. Workstation 2 was delivered to Rome Laboratory in the fall of 1992 after successful completion of Final Acceptance Testing. The delta in RAM and disk space between the two workstations affects performance and the amount of Mentor Graphics software resident on the disk. Workstation 2 contains only a subset of Mentor Graphics Capture Station necessary to run Design Architect.

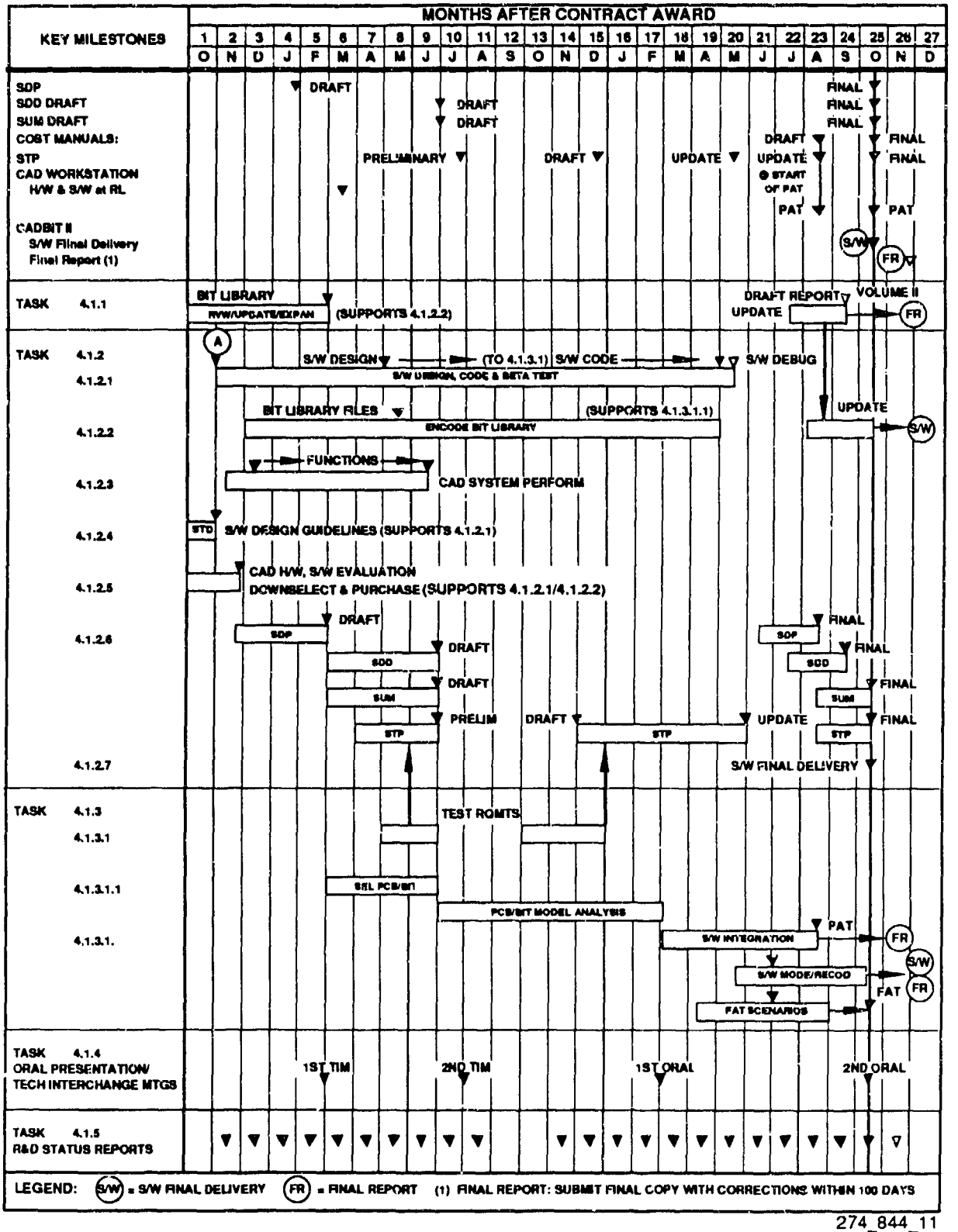
TABLE 2.0 WORKSTATION CONFIGURATION

	Workstation 1	**Workstation 2**
Model	HP 9000 Series 400tC	HP 9000 Series 400tC
Main Memory	32 MB	16 MB
Disk Drives	400MB internal & 1.3 GB external	400 MB internal
Operating System	Domain SR 10.3.5.4	Domain SR 10.3.5
X Window System	X11R4	X11R4
OSF/Motif	V1.1	V1.1
Mentor Graphics	Idea Station V8.1	Capture Station V8.1

2.4 PROGRAM SCHEDULE AND MILESTONES

The CADBIT II Master Program Schedule is shown in Figure 2.2. The main contract tasks (and sections where they are discussed) were:

- Select CAD workstation hardware and software (Sections 3.1 and 3.2).
- Review and update the library of BIT techniques (Section 4).
- Develop and Test a CADBIT II Software module (Sections 3, 4 and 7).
- Document software using DOD-STD-2167A. Four (4) manuals were written during the contract. The manuals are: Software Development Plan, Software Design Document, Software Test Plan and Software User Manual.
- Apply CADBIT II to two (2) Printed Circuit Board testbeds. Two testbeds were selected to validate the CADBIT II System during acceptance testing (Section 8).
- Deliver, install, and demonstrate the CADBIT II System at Rome Laboratory. The CADBIT II System was delivered, installed, and demonstrated at Rome Laboratory October 18-23, 1992.



274_844_11

Figure 2.2 CADBIT II Master Program Schedule

Figure 2.3 shows the milestones met during the contract period.

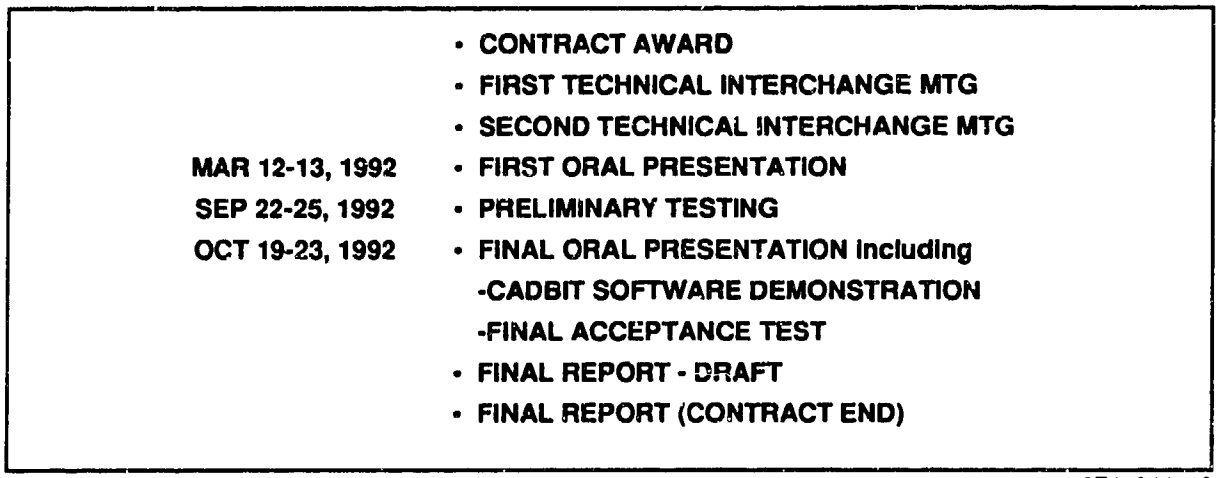


Figure 2.3 Major Contract Milestones

2.5 PROGRAM REVIEWS

Figure 2.4 identifies the occurrences of joint technical meetings held during the CADBIT II contract. Each meeting included a review of contract status, review of draft documentation (if applicable) and delivery of interim products. Table 2.1 shows documentation reviews and interim product deliveries.

REVIEW	WHERE	WHEN
KICKOFF	ROME	NOV 14-15, 1990
1ST TIM*	SAN DIEGO	MAR 5-6, 1991
INTERIM STATUS MTG	ROME	JUN 12, 1991
2nd TIM*	ROME	AUG 13-14, 1991
INTERIM STATUS MTG	SAN DIEGO	NOV 5, 1991
INTERIM STATUS MTG	ROME	DEC 9, 1991
1st ORAL PRESENTATION*	SAN DIEGO	MAR 12-13, 1992
SPECIAL PGM REVIEW	ROME	JUL 29, 1992
SPECIAL TECHNICAL REVIEW	ROME	AUG 4-6, 1992
SPECIAL MGMT REVIEW	ROME	AUG 25, 1992
PRELIMINARY TESTING	SAN DIEGO	SEP 22-25, 1992
FINAL ORAL PRESENTATION* Incl DEMO & FINAL ACCEPTANCE TEST*	ROME	OCT 19-23, 1992

*SOW Requirement

274_844_13

Figure 2.4 CADBIT II Program Reviews

TABLE 2.1 DOCUMENTATION REVIEWS AND INTERIM PRODUCT DELIVERIES

	Documentation	Product
First TIM	SDP, CADBIT I Software Specification redline	Prototype
First informal status		Prototype
Second TIM	SDD, SUM	Prototype, Release 1
First Oral	SUM, SDD, STP	Release 2
Final (Second) Oral	All documentation	Final s/w delivery

3.0 CADBIT II SYSTEM OVERVIEW

The CADBIT II System is composed of one (1) HP 9000 model 400tC CAD workstation, one (1) installed copy of Mentor Graphics Idea Station (Version 8.1) software and the contractor developed software module. The developed software module assists the design engineer in the incorporation of BIT technique(s) into the design of a PCB. The software module operates in an unobtrusive manner, concurrent with Mentor Graphics software. The CADBIT II System performs the following functions: BIT technique selection, insertion, evaluation, on-line help, VHDL file input, and encode BIT library. These functions are explained in Section 5. Figure 3.0 illustrates each piece of the CADBIT II System. The following sections discuss each component of the CADBIT II System.

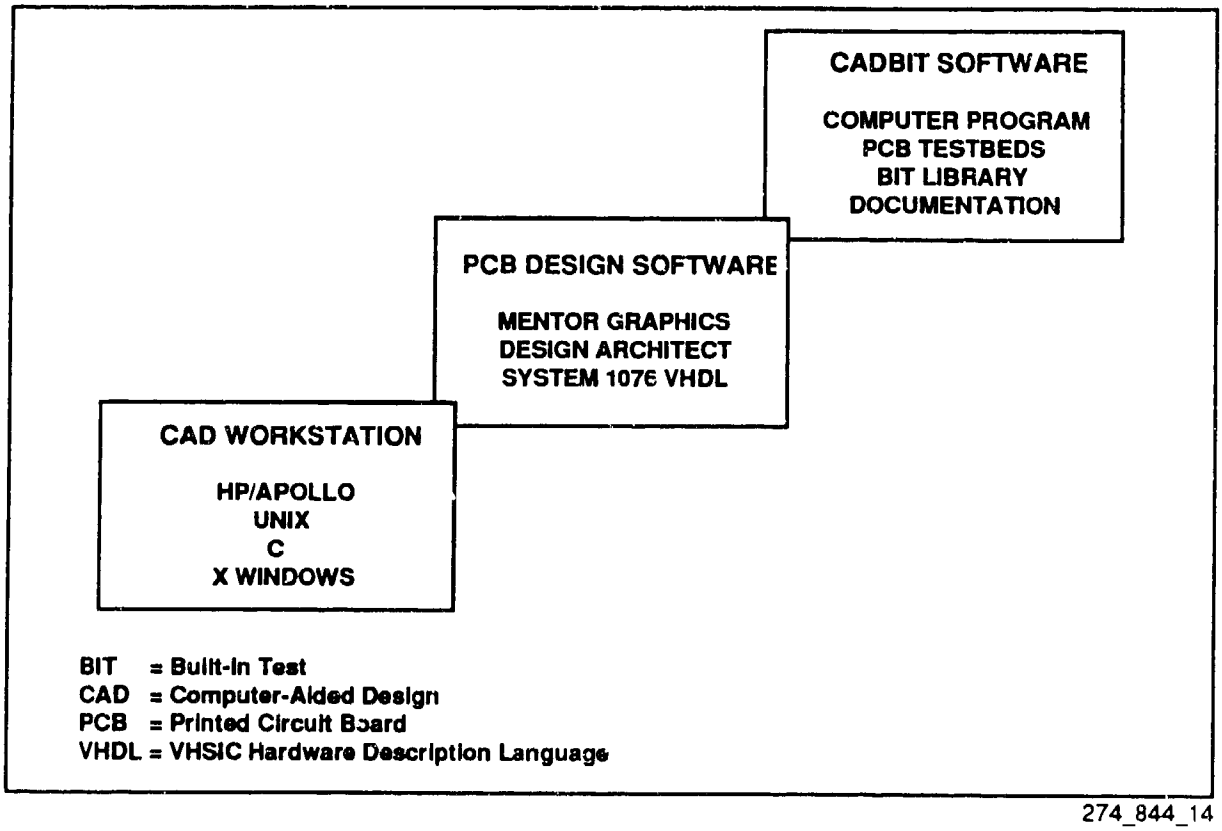


Figure 3.0 CADBIT II System Overview

3.1 PCB DESIGN SOFTWARE

After performing tradeoff analysis of the electronic CAD systems in Table 3.0, Mentor Graphics Idea Station was selected as the CAD system for CADBIT II.

TABLE 3.0 CADBIT SYSTEMS EVALUATED

Vendor	Product
Computervision Corp.	SCHEDIT
Dazix Corp.	DESIGN Entry System
Hewlett-Packard Co.	HP PC Design System
Intergraph Corp.	PCB Engineer
Mentor Graphics Corp.	Idea Station
Valid Logic Systems Inc.	ALLEGRO
Viewlogic	View Draw

Selection criteria included:

OPEN ARCHITECTURE - Mentor Graphics supports industry standards such as VHDL and IGES for file transfer, the Unix operating system and Ethernet/TCP-IP to maximize communications capability with other systems.

SUPPORTING PLATFORMS - Mentor Graphics software is currently available on HP, Apollo, Sun and DECstation workstations.

DATABASE STRUCTURE - Mentor Graphics provides an "open" database structure for queries or updates.

IGES - Mentor Graphics supports the IGES file format so that geometry from other CAD packages can be transferred into the CADBIT II System without requiring that it be redrawn.

EDIF TRANSLATORS - EDIF translators will allow both geometry and the associated netlists to be transferred into and out of the system. This is key in communicating with other electronic CAD systems. EDIF has become essentially a superset of IGES plus netlist capabilities.

VHDL SUPPORT - VHDL provides an Ada-like language for describing the function of electronic designs. When broadly developed in their full form, a VHDL model of the design and EDIF capability will provide seamless design portability between CAD workstations.

Mentor Graphics serves on the committees for EDIF, IGES and VHDL and their products adhere to these standards. Mentor Graphics Idea Station combines Design Architect and QuickSim II with additional capabilities including QuickCheck rules checking and EDIF Netlist Wire interface to export connectivity information. Design Architect is a package for designers that includes schematic capture, VHDL editor, and libraries of digital, analog and ASIC components. QuickSim II is Mentor's advanced simulator and debugger for sub-micron layouts.

3.2 CAD WORKSTATION OVERVIEW

After the selection of the electronic CAD software, a hardware configuration was derived. The configuration shown in Table 2.0 for Workstation 2 was determined to be suitable for the CADBIT II System.

3.3 CADBIT II SOFTWARE OVERVIEW

The CADBIT II software was written using industry standards in C, Unix, X Windows and OSF/Motif. The CADBIT II software was programmed in C, using X and Motif for the user interface, and developed in three incremental builds or phases (Figure 3.1). Each phase extended the previous phase in terms of incorporating additional design information and functionality. The end-product code was fully tested and validated during the acceptance test programs. Section 6 discusses the CADBIT II software in more detail.

3.4 SYSTEM HARDWARE AND SOFTWARE REQUIREMENTS

The CADBIT II System requires use of X Windows, OSF/Motif and Mentor Graphics software. Table 3.1 lists the vendor software and operating systems that served as the baseline for CADBIT II testing. Table 2.2 listed the hardware configuration of the two workstations used during the CADBIT II contract. Mentor Graphics suggests a minimum memory size of 32 MB and a workstation no slower than the 400tCs used by the CADBIT II contract. Disk space should be adequate to hold all software on the system and the CADBIT II System requires 50 MB of disk space.

TABLE 3.1 CADBIT II VENDOR SOFTWARE BASELINE

Software	Version
Domain/OS	SR 10.3.5
X Window System	X11R4
OSF/Motif	V1.1
Mentor Graphics	V8.1

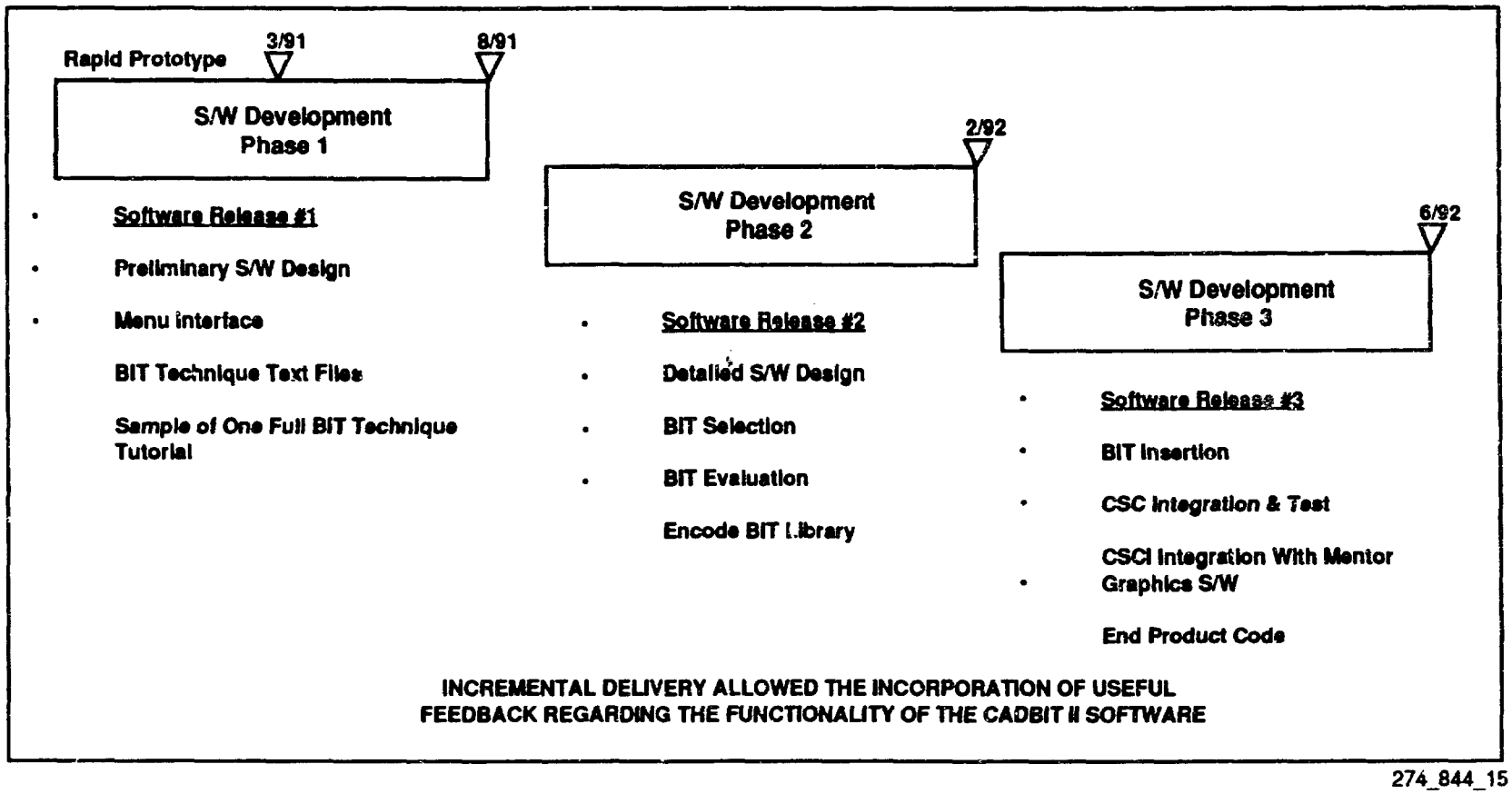


Figure 3.1 CADBIT II Incremental Software Development Process

4.0 BIT LIBRARY OVERVIEW

4.1 REQUIREMENTS

The BIT techniques identified in Volume II of the CADBIT Final Report were reviewed and updated where necessary. The items recognized either missing or incomplete are corrected. Revisions were made to these BIT techniques to make them consistent. Obsolete data and inaccuracies were identified and corrected. Results of this task are documented in Volume II of this report.

4.2 BIT TECHNIQUE DATA

The BIT technique data supports the four basic CADBIT II functions: Tutorial, Selection, Insertion and Evaluation. The BIT technique data is described in detail in Volume II of this report. As illustrated in Figure 4.0, the BIT Library includes text, graphics, and equations data for each BIT Technique as well as "global" data (Parts and Question databases) shared by all techniques.

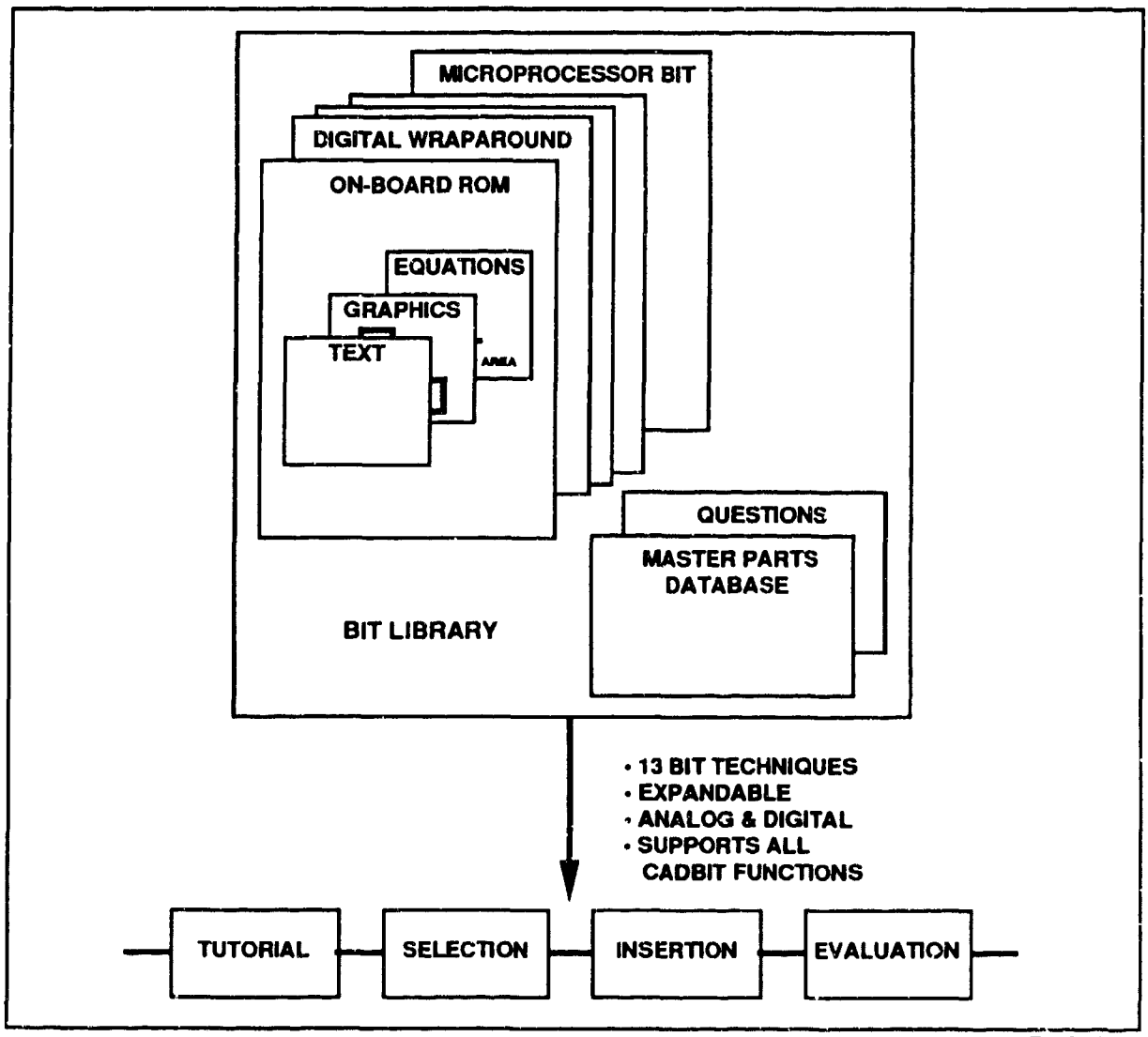


Figure 4.0 BIT Library Supports all CADBIT Basic Functions

4.3 MASTER PARTS DATABASE

The Master Parts Database (MPD) contains information about all components used to implement the BIT techniques available with the CADBIT II software module. It contains all information necessary to support the four basic CADBIT II functions of Tutorial, Selection, Insertion, and Evaluation. For traceability, the database also tracks the sources from which the part information was taken. Table 4.0 lists the MPD elements, provides a brief description of each element, and a sample value using 54ALS253 Selector/MUX as an example sample value.

TABLE 4.0 MASTER PARTS DATABASE ELEMENTS

Database Element and Description	Sample Value
Component ID Component reference identifier.	**U16**
Part Number This is the generic part number for the IC and style number for the resistor and capacitor.	**54ALS253**
Package Type The package type shows how many of the same device in a single package. It can be single, dual, triple, quad, octal, etc.	**QUAD**
Part Description Part name usually describing basic function of component.	**SELECTOR/MUX, 1-OF-2-LINE DATA**
Units per Package This number, usually related to package type above, indicates how many "units" represented by individual CAD symbols are found in a single package.	**2**
Number of Data Bits Data width of the component when applicable.	**4**
Number of Address Bits Address width of memory or microprocessor chips. Not applicable (n/a) for other parts	**n/a**
Number of Pins Total number of physical pins on the component.	**16**

TABLE 4.0 MASTER PARTS DATABASE ELEMENTS, Contd

Database Element and Description	Sample Value
Length Length of the component is shown in inches.	**0.785 inches**
Width Width of the component is shown in inches.	**0.320 inches**
Area per Package Area of the package is calculated by multiplying the Length and Width of the component. The area is given in square inches.	**0.251 square inches**
Typical Power Typical dissipated power of the component shown in milliwatts.	**35.0 mW**
Maximum Power Maximum dissipated power of the component shown in milliwatts.	**77.0 mW**
Weight per Package Weight of the component package in grams.	**2 gms**
Typical delay Typical propagation delay from the device input to output in nanoseconds.	**10 ns**
Maximum delay Maximum propagation delay from the device input to output in nanoseconds.	**30 ns**
Typical Icc Typical power supply current of the component in milliamps.	**7 mA**
Maximum Icc Maximum power supply current of the component in milliamps.	**14 mA**

TABLE 4.0 MASTER PARTS DATABASE ELEMENTS, Contd

Database Element and Description	Sample Value
Nominal VCC Nominal power supply voltage of the component in volts.	**5.0 V**
Maximum VCC Maximum allowed power supply voltage of the component in volts.	**5.5 V**
Reference Source of data extracted for this component. The title and page number of the data book are provided.	**TI ALS p.2-299 (Texas Inst ALS databook)**

4.4 MASTER QUESTION DATABASE

The Master Question Database is a matrix that has a list of questions versus the thirteen techniques using those questions. In the CADBIT II software, the user is expected to answer a merged set of questions corresponding to the suitable BIT techniques. The answers to the questions are captured into variables in the software and are used downstream in component determination equations and penalty equations.

5.0 SYSTEM FUNCTIONS

CADBIT II provides basic designer-oriented functions that assist with selection and implementation of BIT techniques and auxiliary functions to maintain the underlying BIT technique data, provide on-line help, and perform VHDL compilation.

5.1 BASIC FUNCTIONS - OVERVIEW

CADBIT II provides four basic functions to assist the designer with incorporation of Built-In Test in their printed circuit board designs. Figure 5.0 illustrates the four basic functions and the typical sequence in which the designer would use them. This sequence is not mandatory. For example, the Tutorial function is available to designers at any time.

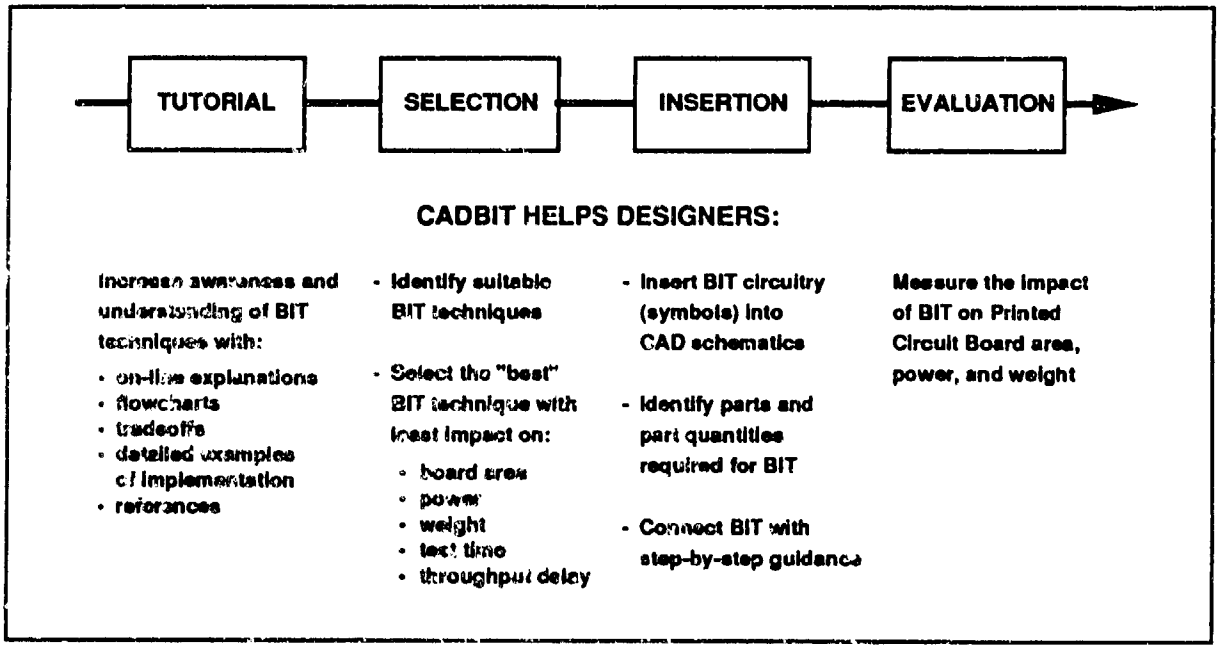


Figure 5.0 CADBIT II Basic Functions

5.2 FUNDAMENTALS

USER REQUIREMENTS

CADBIT is intended for use by designers at an electronic CAD workstation. Basic knowledge of the workstation operating system is required. Knowledge of the electronic CAD software is desirable and is required to use two of the four basic functions (Insertion and Evaluation). It familiarity with the CAD software for effective use by designers.

PRINTED CIRCUIT BOARD (PCB) AND CIRCUIT UNDER TEST (CUT)

Built-in test can be implemented at the system, board, or chip levels. The built-in test domain targeted by CADBIT II is the board level, as shown in Figure 5.1. Techniques in the BIT Library are appropriate for implementation on a single printed circuit board (PCB).

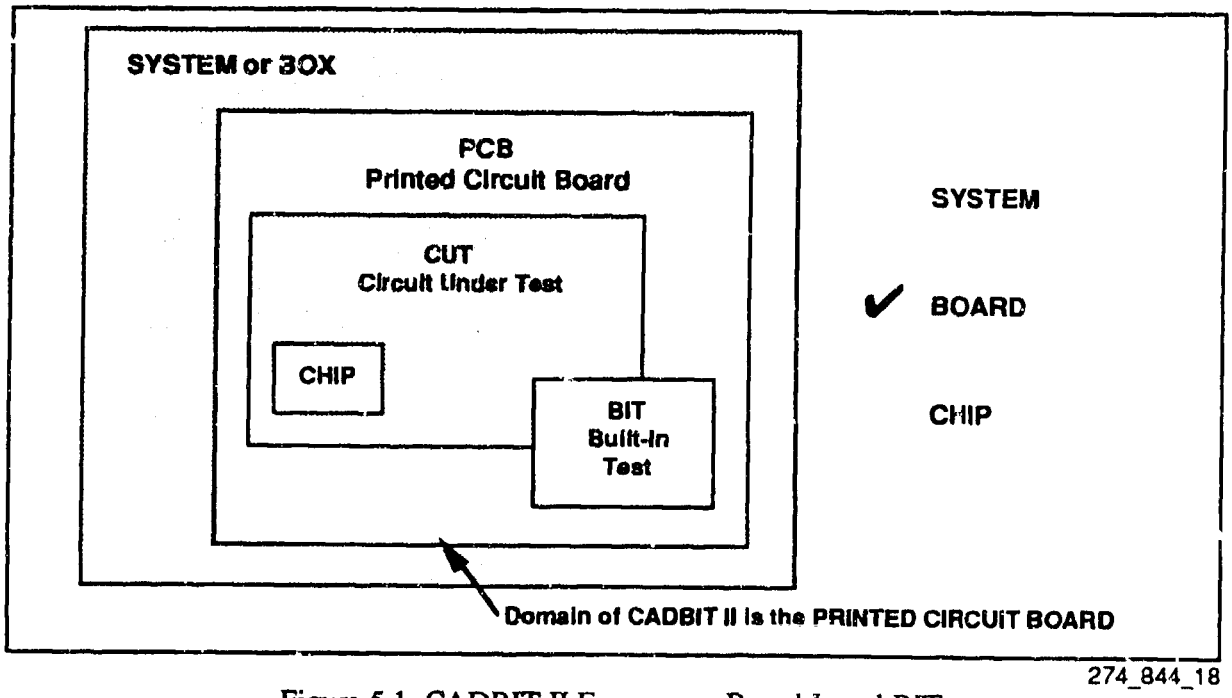


Figure 5.1 CADBIT II Focuses on Board-Level BIT

Designers apply CADBIT II software to a logical subset or partition of the printed circuit board known as the Circuit Under Test (CUT), as illustrated in Figure 5.2. The CUT is an arbitrary partition of the circuit board and may consist of one or more components or chips.

How does the designer identify the Circuit Under Test? Figure 5.2 illustrates the typical process by which designers reach the starting point for CADBIT II application. Generally, the design team will begin by identifying major system-level functions and progressively subdividing these top-level functions until they identify subfunctions suitable for implementation on separate PCBs. At the same time, system-level requirements, e.g. power, weight, size, may also be allocated to each function, subfunction, and board. As shown in Figure 5.2, the board designer "makes the CUT" by further partitioning the board into subfunctions requiring separate BIT techniques.

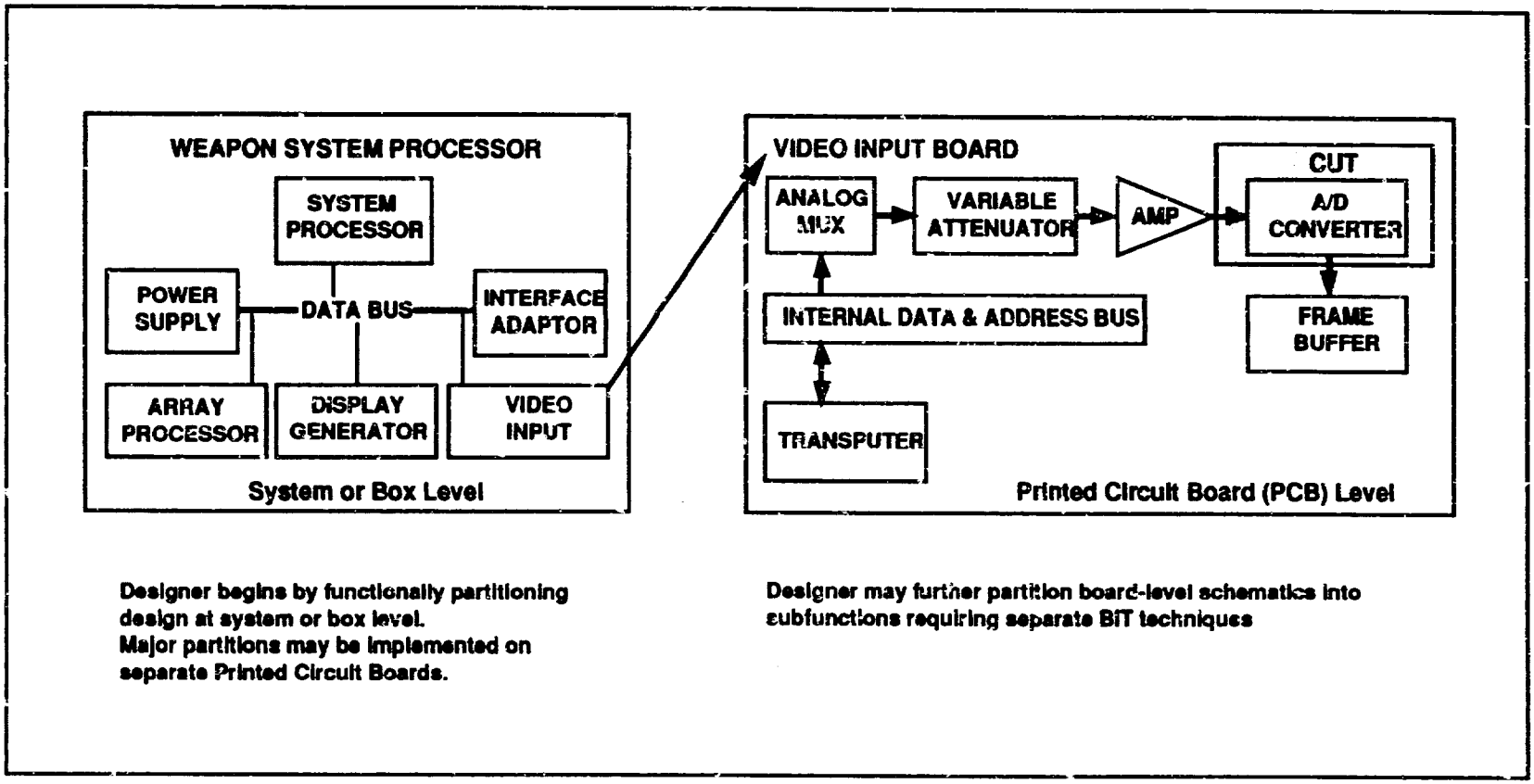


Figure 5.2 How the Designer "Makes the Cut"

BUILT-IN TEST

CADBIT users must be familiar with the basic concepts and goals of BIT. Figure 5.3 below illustrates the anatomy of a typical built-in test. Many variations of this anatomy are possible. In CADBIT II, the focus of the built-in test is the CUT (Circuit Under Test) described above. The CUT will typically have a number of primary inputs or "pins" and a possibly different number of primary outputs. An ideal BIT technique would instantly ensure that the CUT is performing as expected—primary outputs agree with a "Good Machine Response" - for all possible input combinations and sequences.

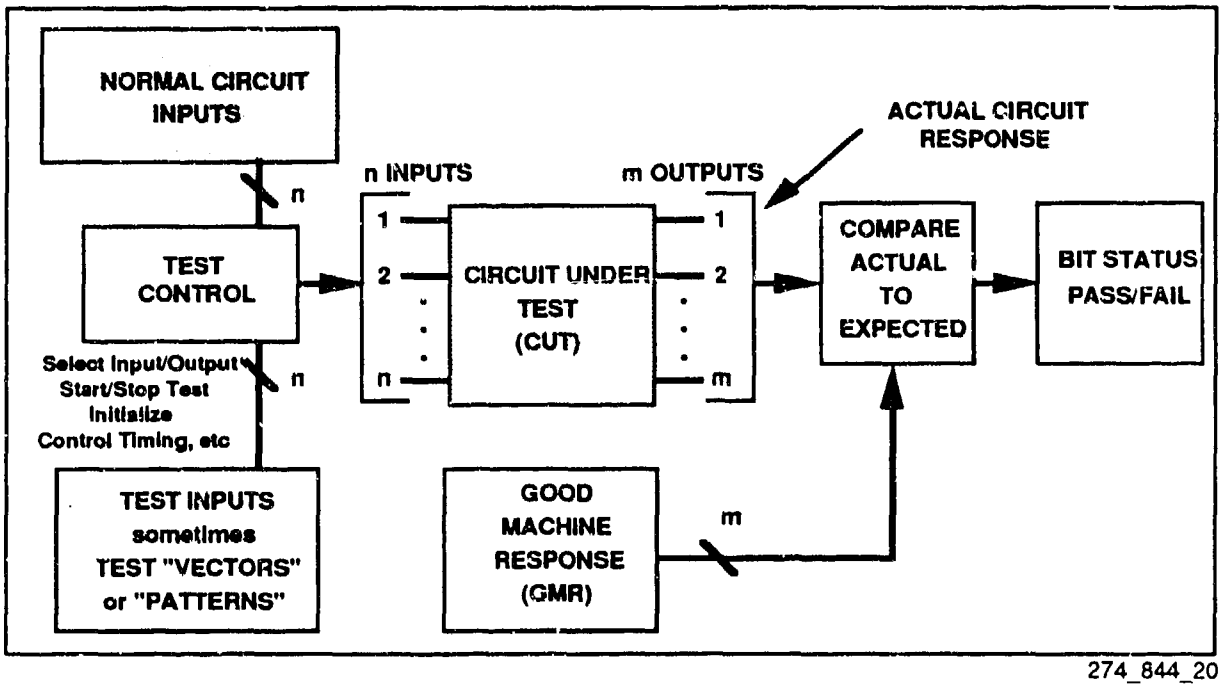


Figure 5.3 Anatomy of a Typical Built-In Test

BIT techniques may need to provide a Test Control function to "disconnect" or alter normal circuit inputs during BIT and "connect" and control delivery of a set of test inputs. For digital circuits, these test inputs are often called test "vectors" or "patterns." The BIT technique must address how these test inputs are generated, stored, and delivered to the Circuit Under Test.

The BIT technique may compare actual circuit response to a known "Good Machine Response" or GMR. Again, the BIT technique must address generation, storage, and delivery of the GMR and may implement the comparison in many different ways.

Finally, the BIT technique needs to provide some indication of the results of the test. As a minimum, the BIT Status function should indicate whether the test passed or failed. The BIT status function may also isolate detected faults to specific components and provide diagnostic information to system operators and maintainers.

5.3 TUTORIAL

The CADBIT II Tutorial function provides designers with an extensive database of BIT technique information. This BIT technique information can be viewed by designers on-line at the CADBIT workstation. Figure 5.4 describes the important features of this function and the type of data provided.

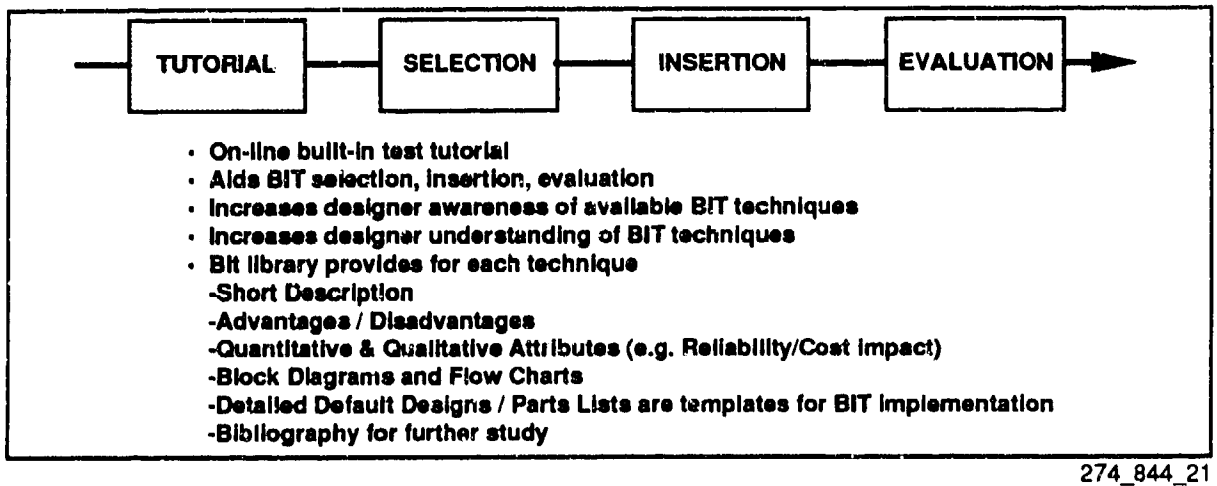


Figure 5.4 Overview of BIT Technique Tutorial Function

Complexity and variation in the design process make total "push-button" automation of BIT technique selection unadvisable. CADBIT II software is not intended to replace engineering judgment. The Tutorial function plays an important role supplementing the functions of Selection, Insertion, and Evaluation by providing designers additional qualitative and quantitative information that serve to educate and strengthen designer judgment.

The Tutorial function provides designers with access to data in the CADBIT II "BIT Library" discussed in Section 4. Figure 5.5 illustrates the data elements in the BIT Library available for on-line inspection and indicates which CADBIT II functions are supported by the data. It should be noted that some BIT Library data elements support more than one of the major functions. Access to some of the BIT Library data (for example Equations and Parts Data) is accomplished through the Utilities option in the CADBIT Main Menu.

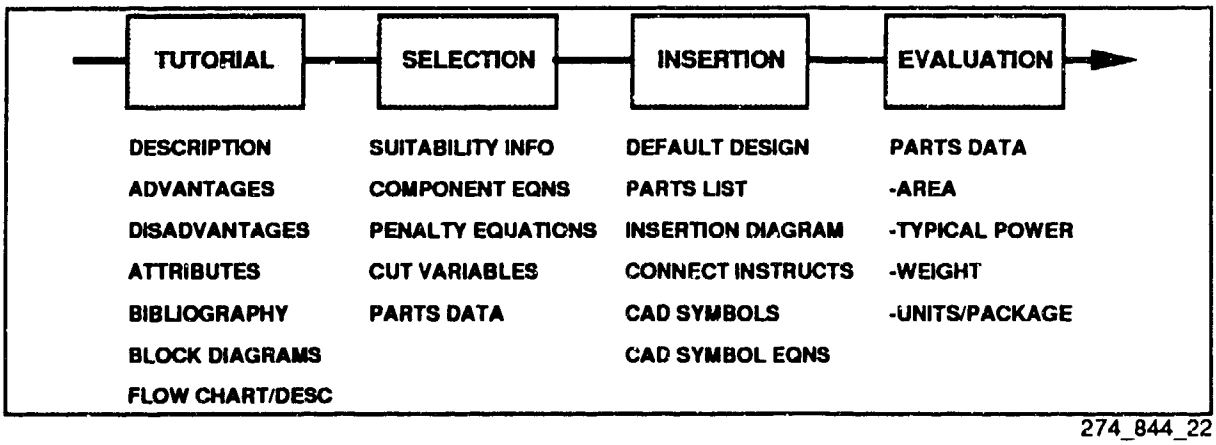


Figure 5.5 The Tutorial function provides on-line access to BIT Library data used to support all major CADBIT functions.

5.4 SELECTION

The Selection function helps designers choose the "best" BIT technique for their Printed Circuit Board (PCB) designs. The Selection function has two major steps. CADBIT II first helps the designer determine which BIT techniques are suitable and then ranks the suitable techniques to help the designer find the "best" one. Figure 5.6 shows the important features of the Selection function.

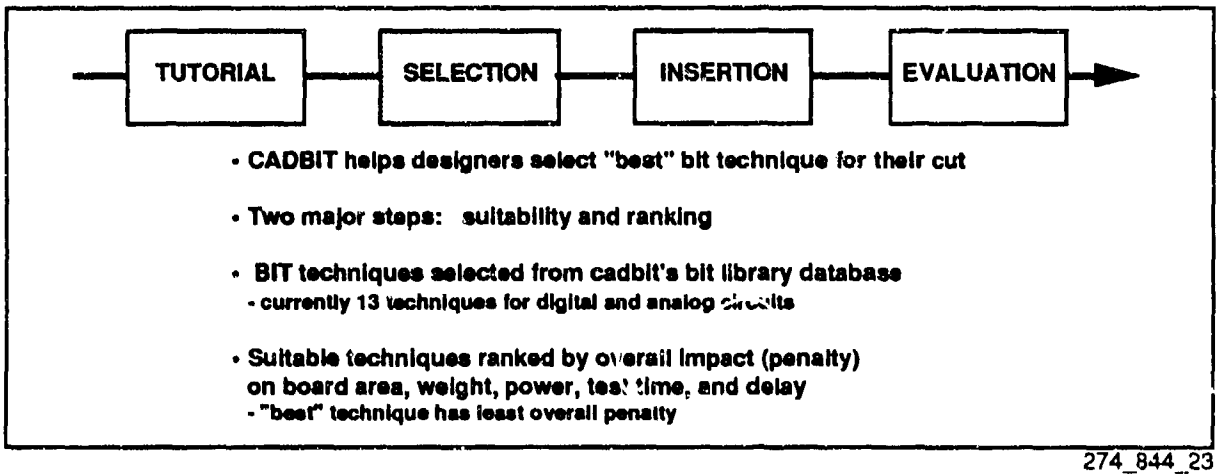


Figure 5.6 Overview of the BIT Selection Function

There are many possible definitions of "best" technique. CADBIT II defines the "best" technique as the one which minimizes a combination of 5 design penalty areas.

CADBIT II provides for user control and flexibility in the Selection process. As discussed above in section 5.4, engineering judgment and the BIT Tutorial information are intended to supplement the selection decision process. Users may examine the reason for technique deselection and have override control of suitability determination. That is, designers have the option to "force" a BIT technique to be "suitable" or "unsuitable." Users control the weighting factors used to combine individual penalty scores for area, power, weight, test time and throughput delay. Finally, users control the final selection from the ranked list of BIT techniques provided by CADBIT II.

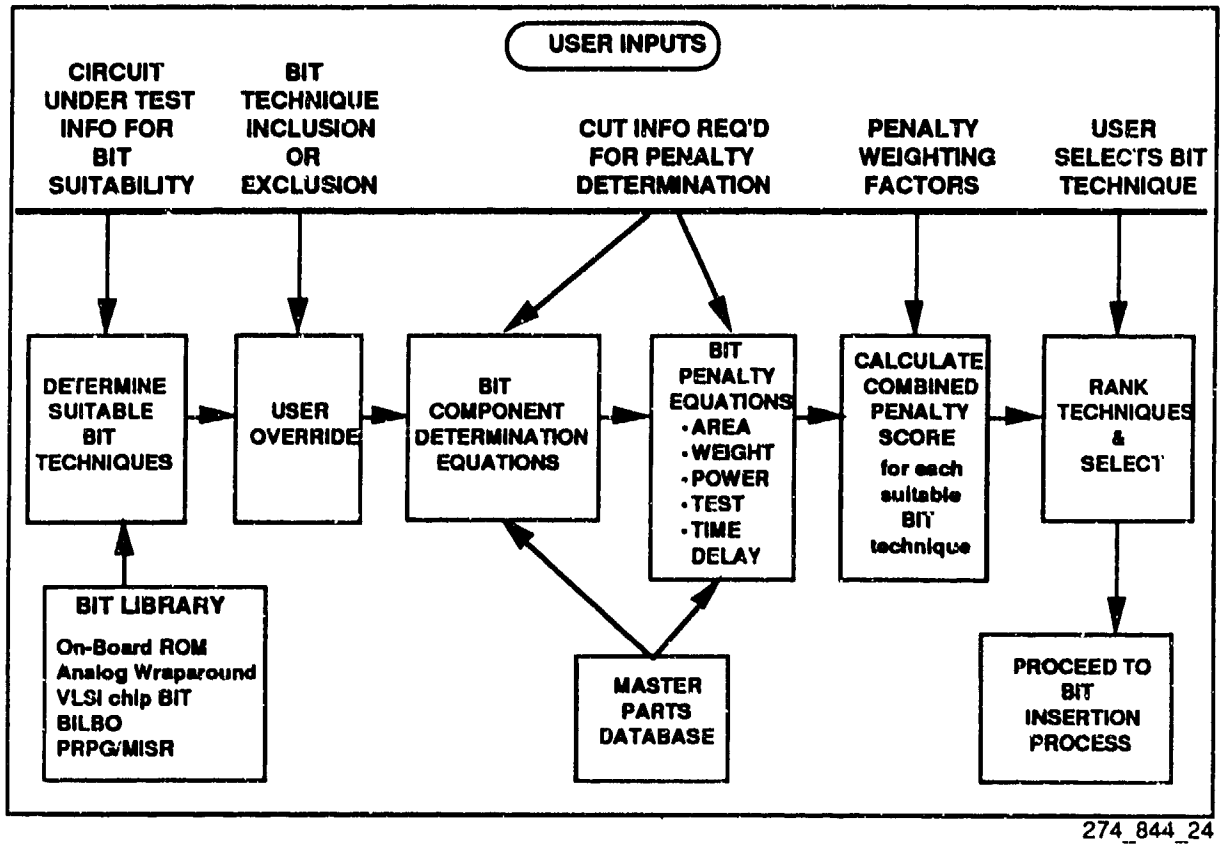


Figure 5.7 BIT Selection Process Flow

Figure 5.7 shows a roadmap of the CADBIT II BIT Selection process. The roadmap has three tiers of information. At the top, the user's role in the process is highlighted. The middle section shows the logical sequence of steps in the CADBIT II Selection process. The bottom section shows data structures internal to the CADBIT II software and how they provide data to the CADBIT II processes.

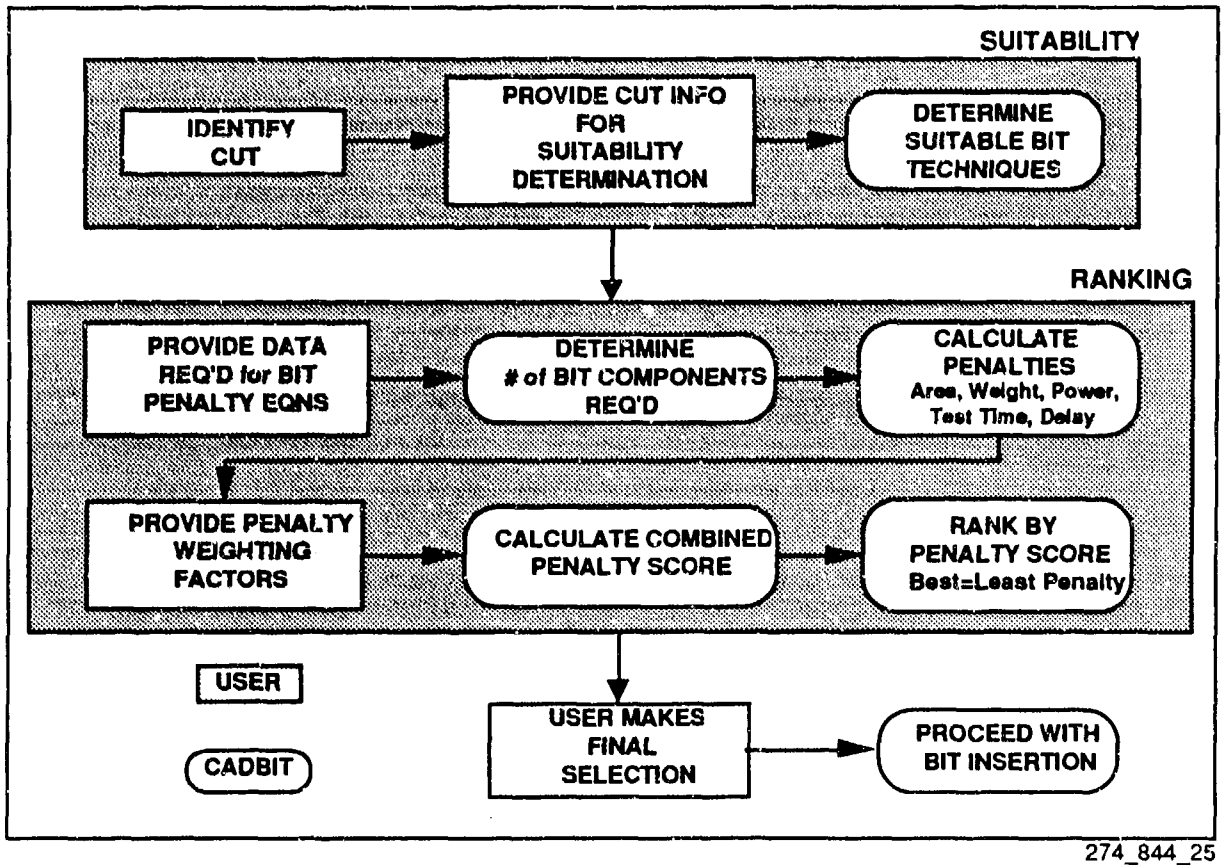


Figure 5.8 CADBIT II BIT Selection Process Steps

Figure 5.8 shows the Selection process in a step-by-step format. Actions or data required from the user are indicated by shaded boxes. Actions automated by CADBIT II software are shown as unshaded, rounded boxes. As discussed above, the Selection process activities can be grouped into two major subprocesses: Suitability and Ranking.

SUITABILITY

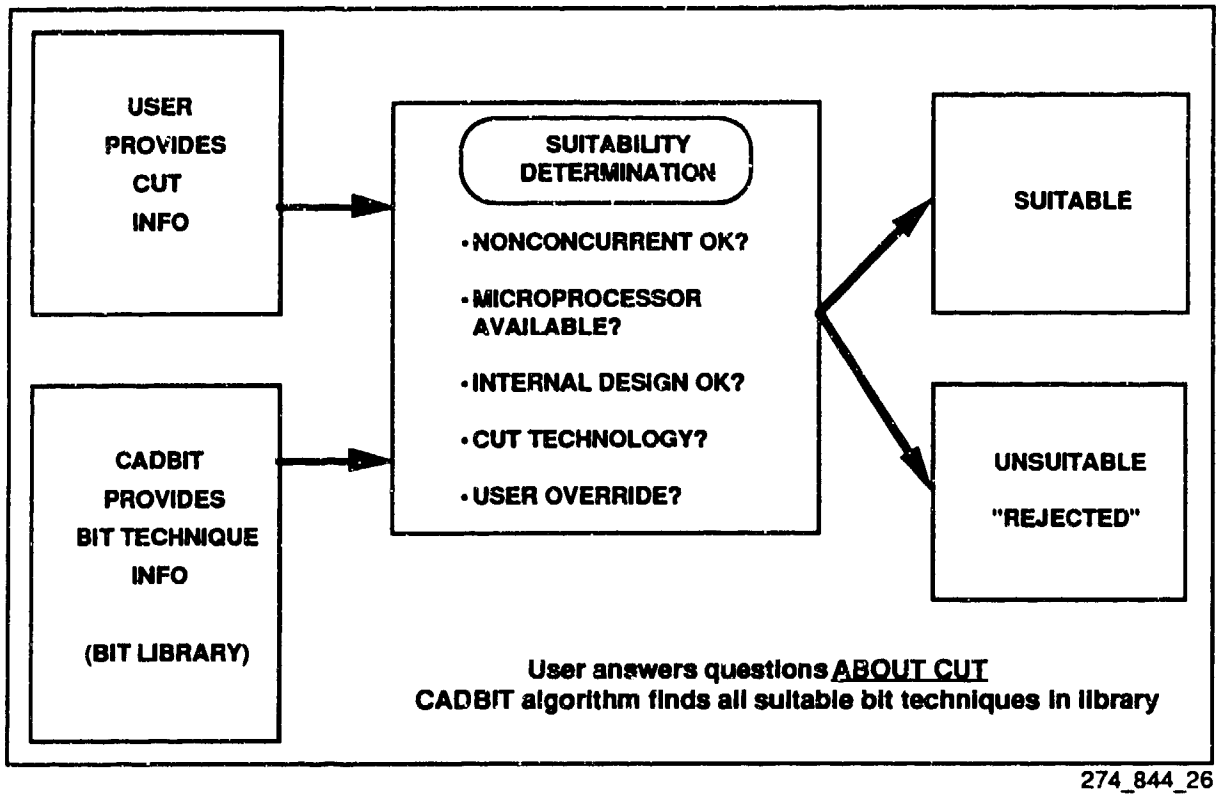


Figure 5.9 How CADBIT II Determines BIT Technique "Suitability"

Figure 5.9 illustrates how the Suitability function determines which BIT techniques are suitable for the designer's particular Circuit Under Test (CUT). All techniques in the BIT Library are considered and classified as either Suitable or Unsuitable. The algorithm functions much like a computerized dating service comparing user input about the CUT with associated information about each BIT technique in the BIT Library. The user may override the Suitability algorithm and "force" any BIT technique to be Suitable or Unsuitable.

Users must answer four questions about their CUT:

1. WHAT IS THE CUT COMPONENT TECHNOLOGY?
2. IS A NONCONCURENT BIT OK?
3. IS A MICROPROCESSOR AVAILABLE FOR BIT?
4. IS INTERNAL CUT DESIGN OK?

The following paragraphs explore each of the four Suitability tests, the meaning of the questions, the possible user responses, and the consequences of each response on the Suitability determination algorithm.

CUT TECHNOLOGY TEST

Designers must indicate the technology of components in their CUT (not including any BIT circuitry to be incorporated). Figure 5.10 indicates the responses recognized by CADBIT II and the consequences of those responses on BIT technique suitability determination.

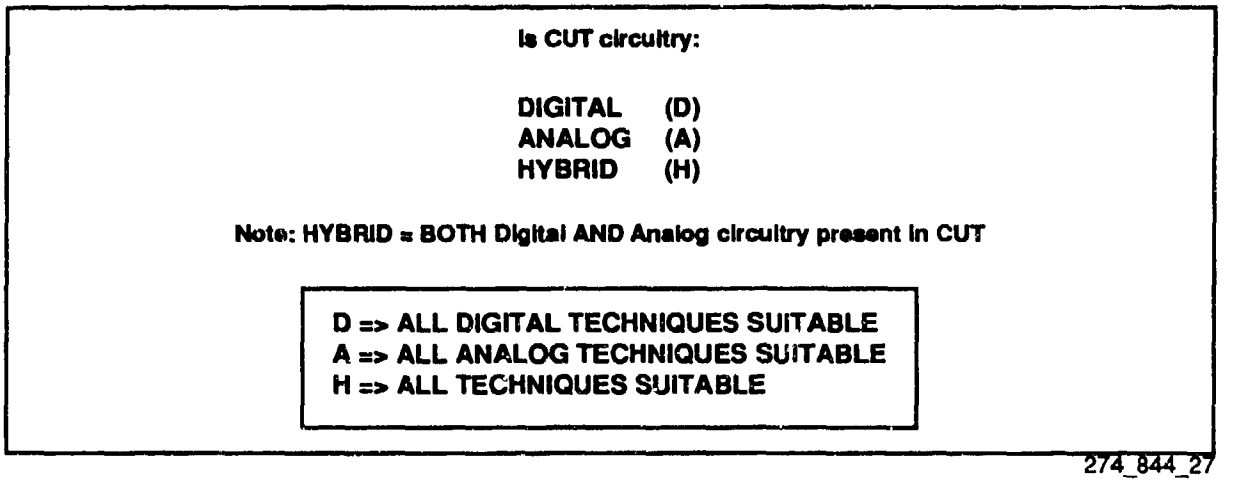


Figure 5.10 CUT Technology Test for BIT Suitability

The technologies recognized by CADBIT II are:

Digital	(D)
Analog	(A)
Hybrid	(H)

The CUT technology is "Digital" if ALL components in the CUT are digital, "Analog" if ALL components are analog, and "Hybrid" if BOTH digital and analog components are present.

Each BIT technique in the BIT Library has been analogously classified as *applicable to* Digital, Analog, or Hybrid CUTs. It should be noted that the BIT technique classification does NOT refer to the technology of the components used to implement the technique but rather the technology of the CUTs typically tested with the technique.

If the user indicates the CUT is Digital, then only the Digital techniques in the BIT Library will be considered for suitability. Analog and Hybrid techniques will be rejected as unsuitable. Similarly, if the user indicates the CUT is Analog, then only the Analog techniques in the BIT Library will be considered. Digital and Hybrid techniques would be rejected. Finally, if the user indicates the CUT is Hybrid, then the presence of BOTH digital and analog circuitry in the CUT means that all techniques in the BIT Library may be suitable (if they pass all other Suitability tests). In this case, no techniques would be rejected as a result of the CUT Technology test.

CONCURRENCY TEST

Designers are asked whether it is OK to test their CUT with a nonconcurrent BIT technique. Figure 5.11 indicates the consequences of "Yes" and "No" responses on BIT technique suitability.

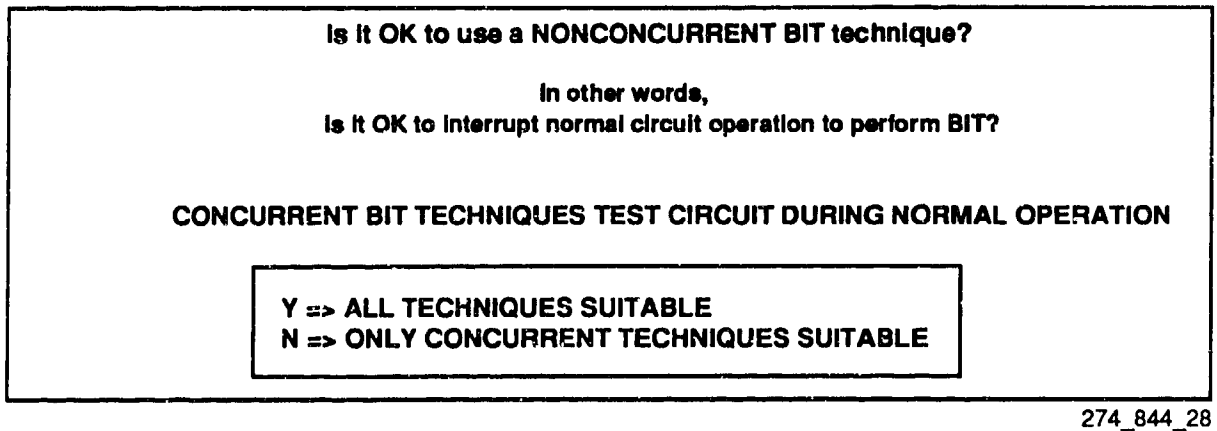


Figure 5.11 Concurrency Test for BIT Suitability

This question is asking the user if it is OK to interrupt normal circuit operation (usually for a very short time period) to perform testing. A nonconcurrent BIT technique interrupts normal circuit operation during BIT. Concurrent BIT techniques can test the circuit using normal operational inputs and so can be performed without interrupting normal circuit operation.

A "Yes" response means that all techniques in the BIT Library may be suitable (if they pass all other Suitability tests). No techniques will be rejected as a result of the Concurrency test. If the user response is "No," then nonconcurrent BIT techniques are rejected and only the concurrent BIT techniques in the BIT Library are further considered for suitability.

MICROPROCESSOR AVAILABILITY TEST

Designers are asked whether a microprocessor is available for BIT. Figure 5.12 indicates the consequences of "Yes" or "No" responses on BIT technique suitability.

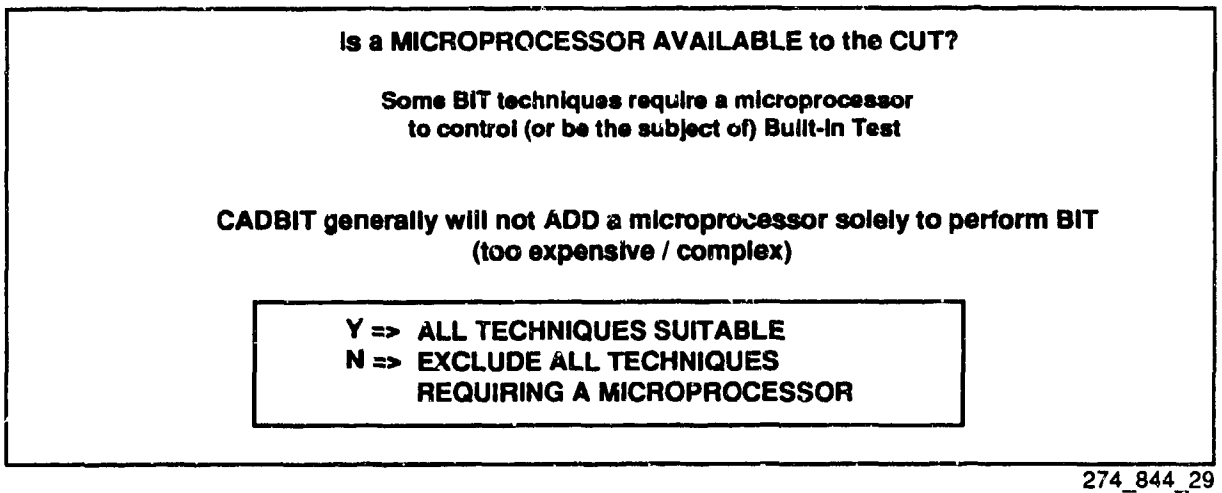


Figure 5.12 Microprocessor Availability Test for BIT Suitability

The CADBIT II suitability algorithm will generally not add a microprocessor for the sole purpose of performing BIT. This approach is considered too expensive and complex. Exceptions to this rule are made when the BIT technique requires exclusive use of the microprocessor (e.g. the "On-Board Integration of VLSI Chip BIT" technique).

There are two aspects of this question that need to be considered by the user. First, the question is asking whether a microprocessor is already in the CUT or available to the CUT. Second, the question is asking whether the existing microprocessor can be shared (i.e. has sufficient reserve capacity) to control, provide storage resources, or be the subject of BIT.

Existing microprocessors with reserve capacity are not considered to contribute to area, weight, or power penalties. Where microprocessors are exclusively used for BIT, their area, weight, and power requirements are reflected in the design penalties. If the user indicates that a microprocessor is available to the CUT for BIT, all techniques may be suitable. No techniques will be rejected as a result of the Microprocessor Availability test.

INTERNAL CUT DESIGN TEST

Designers are asked whether it is OK to modify the CUT internal design in order to implement a BIT technique. Figure 5.13 indicates the consequences of "Yes" or "No" responses on BIT technique suitability.

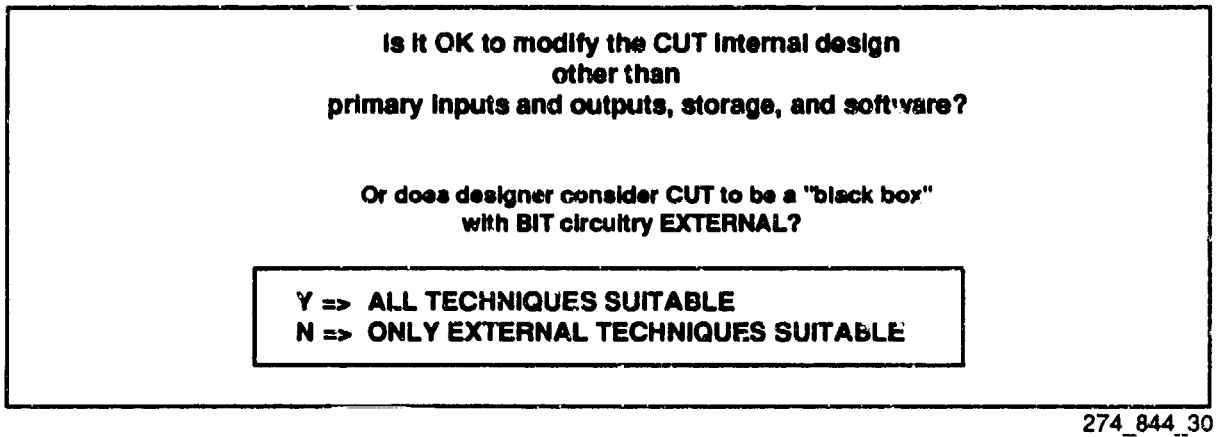


Figure 5.13 BIT Suitability Internal Design Test

Many BIT techniques can be implemented "externally" to the CUT, requiring access only to primary input and output pins. Non-external BIT techniques require access to or modification of the internal design of the CUT A "No" response allows designers to indicate that their CUT should be considered a "black box" with no internal modifications allowed. There are some grey areas in this Internal/External categorization. CADBIT II does not consider sharing memory or modification of software within the CUT to be automatic grounds for unsuitability. If the user indicates that internal CUT design is OK, all techniques may be suitable. No techniques will be rejected as a result of the Internal Design test. If the user indicates that internal CUT design is not OK, BIT techniques requiring internal design modifications are rejected.

AUTOMATIC SUITABILITY DETERMINATION

The CADBIT II Suitability algorithm consists of four tests: CUT Technology, Concurrency, Microprocessor Availability, and Internal CUT Design, as discussed above. Figure 5.14 shows how the Suitability algorithm is applied to each BIT technique in the BIT Library to determine whether the technique is suitable or rejected from further consideration. CADBIT II also permits users to inspect the reason for technique rejection (i.e. which of the four Suitability tests failed).

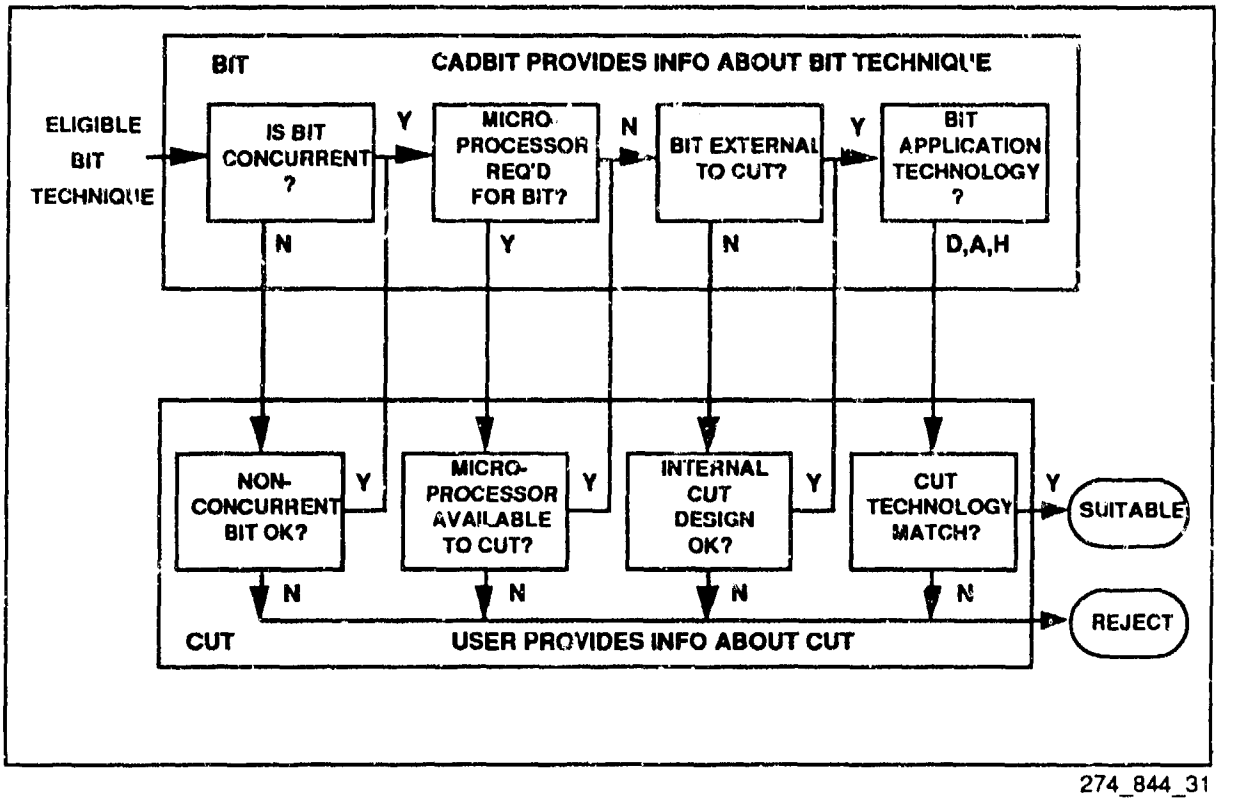


Figure 5.14 CADBIT II Suitability Algorithm

USER OVERRIDE OF SUITABILITY DETERMINATION

Designers may override automatic suitability determination, forcing individual techniques to be considered "suitable" or "unsuitable." The ability of the designer to exercise engineering judgment is an important feature of the CADBIT II software design. For example, we have seen that CADBIT II does not consider memory sharing or software modification within the CUT to be grounds for unsuitability in the Internal Design test. In circumstances where this type of impact is unacceptable, the designer has the flexibility to force offending techniques into the Unsuitable category.

RANKING

After the Suitability process has identified all BIT techniques in the BIT Library that are suitable for a designer's Circuit Under Test (CUT), CADBIT II ranks the suitable techniques by an overall "design penalty" score. The Ranking process estimates the adverse impact of each suitable BIT technique on board area, weight, power, test time, and throughput delay. The overall penalty score is a linear combination of the five design area penalties. Figure 5.15 provides an overview of the Ranking process.

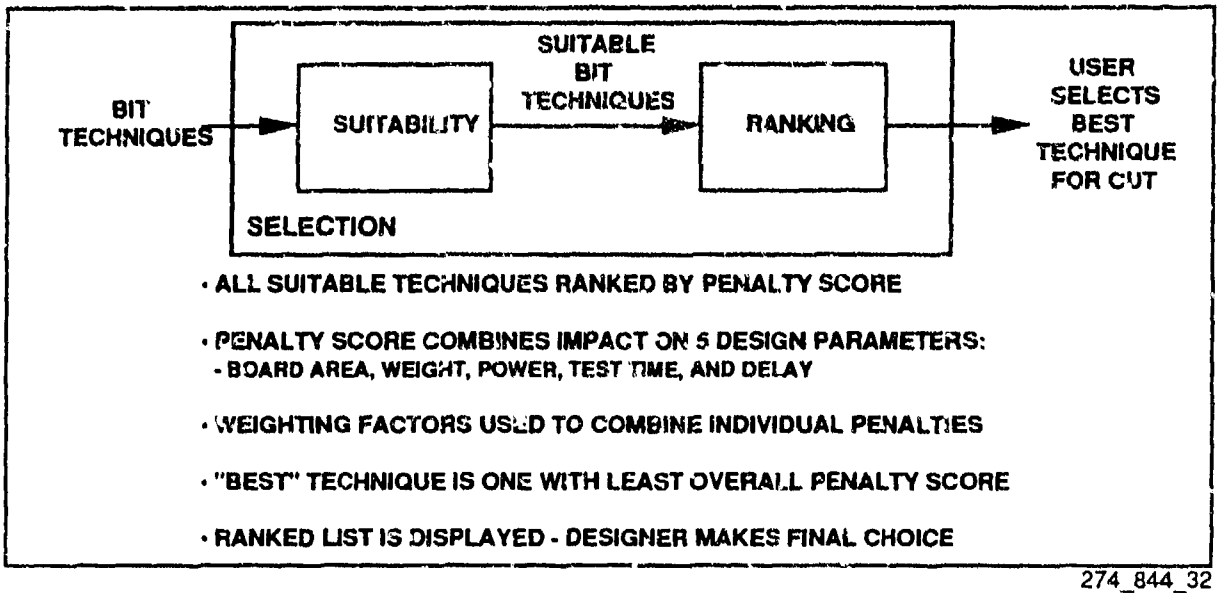


Figure 5.15 Overview of the BIT Technique Ranking Process

OVERALL PENALTY SCORE

Figure 5.16 illustrates the method used by CADBIT II to calculate an overall penalty score used to rank the suitable BIT techniques. Since adding BIT generally adversely impacts (increases) board area, power, weight, test time, and throughput delay, techniques are ranked by "penalty" score -- the "best" technique has the "lowest" penalty. The Penalty Weighting Factors used to combine the individual penalties are controlled by the designer. The weighting factors indicate the relative importance to the designer of mitigating impact on each of the five design parameters. Designers should set the weighting factors to reflect design priorities and concerns specific to the designer's CUT and the board or system containing the CUT.

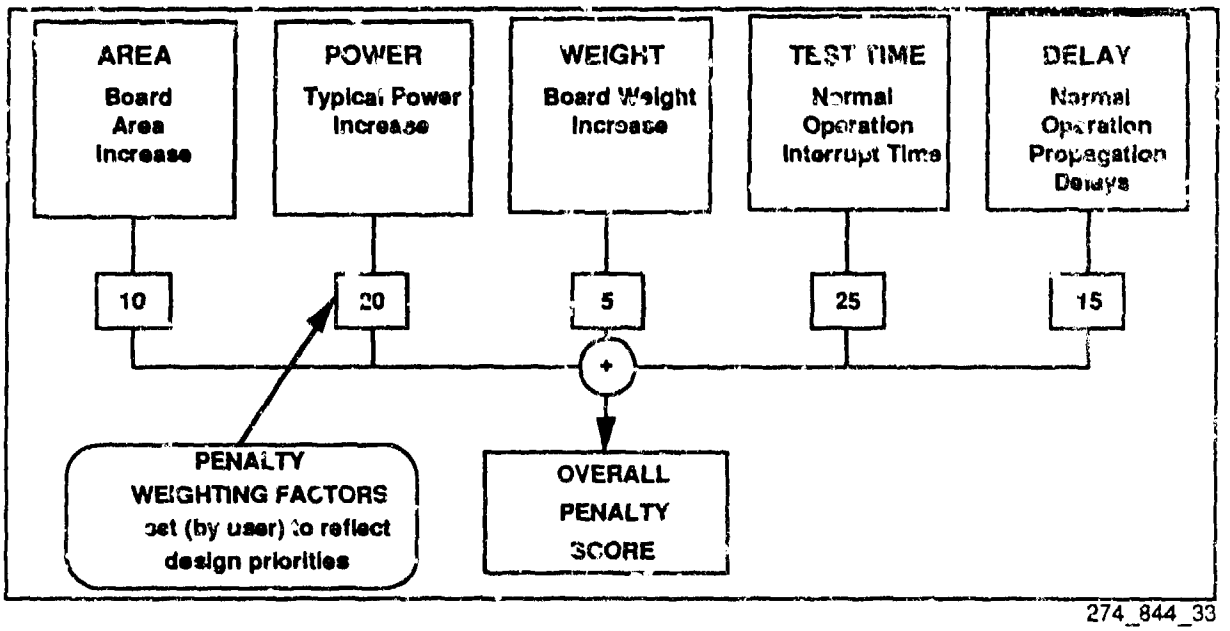


Figure 5.16 Overall Penalty Score

INDIVIDUAL PENALTY SCORES

CADBIT calculates individual penalty scores using Penalty Equations from the BIT Library. Each technique has its own set of Penalty Equations for each of the five penalty parameters.

AREA, POWER, AND WEIGHT PENALTIES

Area, Power, and Weight penalties are directly related to the additional components added to implement a BIT technique. To calculate these penalties, CADBIT II must identify the parts and part quantities needed to implement each BIT technique. The part quantities are calculated using Component Determination Equations (CDEs) in the BIT Library database. These CDEs in turn are functions of certain CUT variables, such as "number of CUT inputs" or "number of CUT outputs." The Power penalty score measures the increase in typical power required rather than maximum power since designers surveyed felt this was a more relevant design parameter.

A typical Penalty Equation for Area would look like:

$$\text{Area Penalty} = \text{Area of components and traces to implement BIT}$$
$$= n_1 * a_1 + n_2 * a_2 + \ldots + n_i * a_i + \ldots + 15\% \text{ for traces}$$

Where:

n_i = Number of i-th part type required to implement BIT
a_i = Area of i-th part type

The CDEs calculate the part quantities ("n_i"s) in terms of CUT variables such as "number of CUT outputs." The part areas ("a_i"s) for specific parts are looked up in the BIT Library's Master Parts Database which contains information on all parts used to implement the BIT techniques in the BIT Library Section 4 discusses the BIT Library and Master Parts Database in more detail.

TEST TIME PENALTY

The Test Time penalty measures the period of time that normal circuit operation must be interrupted to perform the test. We have seen that for "concurrent" BIT techniques, no interruption to normal circuit operation is required, so for these BIT techniques, the Test Time penalty is zero. Nonconcurrent techniques suspend normal circuit operation during the test usually disconnecting or altering the normal circuit inputs in favor of more demanding test inputs.

It should be noted that some concurrent BIT techniques do have a "test time" required to complete the test. Since the concurrent BIT takes place in parallel with normal circuit operation, it is not considered to adversely impact the design and CADBIT II does not include this time in the Test Time penalty score.

THROUGHPUT DELAY PENALTY

The "Delay" penalty measures the increased signal propagation delays during normal circuit operation introduced by added BIT circuitry and software. The delays usually result from components added to implement a BIT technique that are in series with the normal signal path. For example, a multiplexer used to select between normal circuit inputs and test inputs would add to throughput delay during normal operation.

USER REQUESTED DATA

CADBIT II requires additional information about the CUT to calculate individual penalty scores. This information, known to CADBIT II as "User Requested Data" (URD), must be provided by the designer and represents the most demanding portion of the CADBIT II user interface. The User Requested Data is solicited from designers by prompting them with a series of questions. As discussed above, the penalty equations for area, power, and weight depend on part quantities calculated using Component Determination Equations (CDEs). The Penalty and Component Determination Equations for each technique are expressed in terms of CUT parameters or variables. Different BIT techniques may require different CUT variables in their equations. Since the user must provide the CUT information, increasing the number of CUT parameters in the Penalty and Component Determination equations increases user burden. Streamlining and standardizing the CUT parameters used in the equations increases user acceptance. Fortunately, the CADBIT II team was able to significantly streamline and standardize the data requirements, resulting in fewer questions and more commonality in CUT parameter usage. Figure 5.17 shows an overview of the User Requested Data process.

On-line help is available to help designers answer the User Requested Data questions. In particular, the on-line help identifies the BIT techniques which require the information being requested. If the User Requested Data is inappropriate for the CUT, designers can exercise the Suitability override option and force the BIT techniques giving rise to the information request to be rejected as unsuitable.

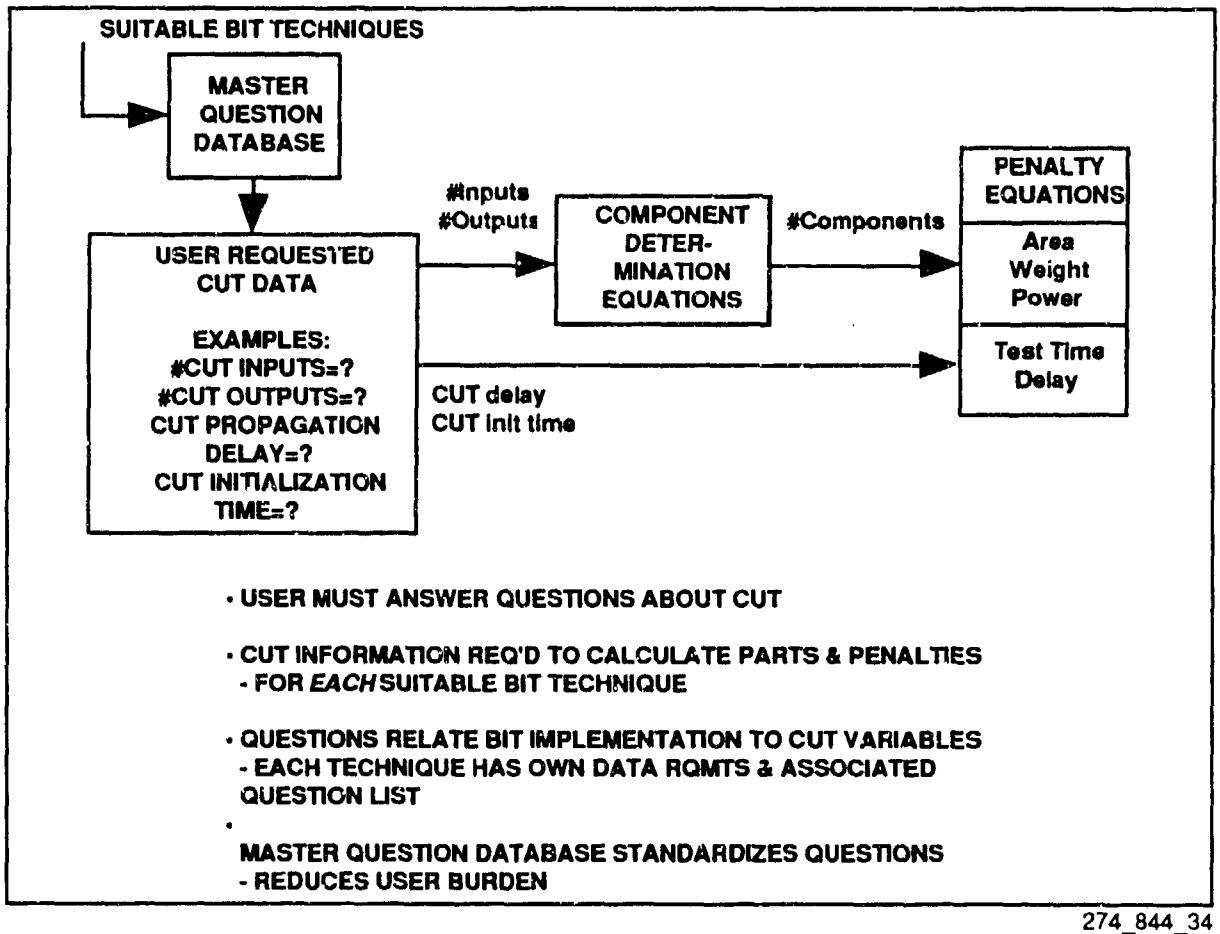


Figure 5.17 Users must provide CUT data to calculate penalty scores.

COMPONENT DETERMINATION EQUATIONS

Component Determination Equations (CDEs) help CADBIT II determine the part quantities required to implement a BIT technique. The CDEs also support the Insertion and Evaluation functions.

The Penalty Equations and the CDEs are based upon a BIT technique's "Default Design" maintained in the BIT Library. The Default Design is a detailed wiring diagram identifying specific parts and connections. The Default Design is useful as a template for BIT technique implementation. CADBIT II tailors the Default Design to specific CUT requirements. For example, the Default Design is "scaled" by the CDEs to the "size" of the CUT. That is, the CDEs will increase or decrease the number of components required for BIT implementation depending on the "number of CUT inputs" and the "number of CUT outputs."

The concept of CDEs and their dependence on CUT variables supplied by the user (see User Requested Data, above) is best understood through example. Figure 5.18 shows how a typical CDE depends on CUT variables (in this case, "number of CUT outputs") and illustrates the concept of "scaling" the Default Design.

A "comparator chip" is commonly used to implement the compare function when the circuitry involved is digital. The Default Design identifies a specific chip from a specific technology family. The technology family used by CADBIT II for most digital chips is ALS (Advanced Low-Power Schottky) since it represents a modern technology that offers a reasonable balance of (low) power and speed.

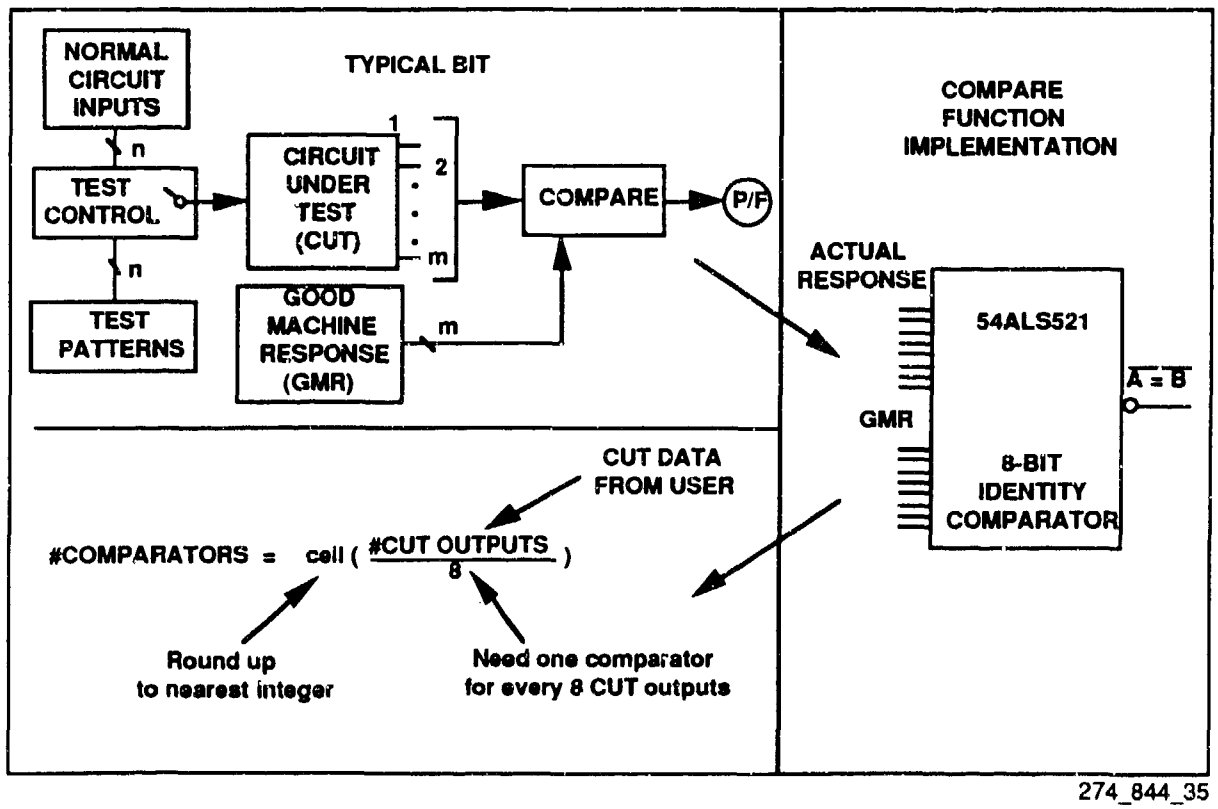


Figure 5.18 Typical CDE is a function of CUT data supplied by users.

In ALS technology, CADBIT II implements the Compare function using the 54ALS521 8-bit identity comparator chip. Once a specific chip has been identified, the relationship to the CUT becomes clearer. As shown in Figure 5.18, the 54ALS521 chip compares signals on two sets of 8 input pins and indicates the result of the comparison on a single output pin. A typical BIT technique would use one set of 8 pins for the Actual Circuit Response and the other set for the Good Machine Response. Rather than limit the CADBIT II application to CUTs with 8 or fewer outputs to check, the CDEs scale the Default Design by asking the user how many CUT outputs the designer needs to test. For every 8 CUT outputs, the CDEs will add another comparator chip.

COMPONENT PACKAGING

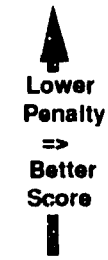
The CADBIT II team identified and resolved a problem with the CDEs involving real-world constraints of component packaging. Components are not always sold and used as individual units but are sometimes multiply bundled in component packages. For example, the Hex Inverter package, 54ALS04B, used by CADBIT II actually contains 6 inverters. If 1-6 inverters are required for BIT, only one Hex Inverter package is required. Packaging constraints were accounted for by including "units per package" in the Master Parts Database for each part.

Accounting for component packaging is important to accurately calculate the Penalty Equations, especially for area, weight, and power. A BIT technique requiring 3 inverters should not be overpenalized by including three times the area, weight, and power of the hex package since only one package is required. On the other hand, the BIT technique would be underpenalized if the area penalty included only three times the area of each inverter "unit," since, in real-world board layout, the entire package of six would be mounted on the board.

BIT TECHNIQUE PENALTY REPORT

Figure 5.19 illustrates how CADBIT II ranks the suitable BIT techniques and displays a "BIT Technique Penalty Report" showing the results of the individual and overall penalty equation calculations. The designer may select any BIT technique in the list and proceed with the Insertion function described in the next section.

RANK	SUITABLE BIT TECHNIQUES	PENALTIES					Penalty Score	
		Area (sq.in.)	Typical Power (mW)	Weight (gms)	Test Time (µsec)	Delay (ns)		
1.	Error Correction Codes	17.00	521.0	25.0	0	67	10	
2.	Microprocessor BIT	19.50	541.3	90.2	1520	120	15	
3.	On-Board ROM	6.64	312.2	39.2	310	38	22	
4.	Digital Wraparound	1.38	213.8	10.5	210	19	32	
5.	Analog Wraparound	5.82	1968.0	42.2	180	17	40	
	Penalty Weighting Factors	10	25	5	20	20		



274_844_36

Figure 5.19 Suitable BIT techniques are ranked by penalty score.

5.5 INSERTION

Figure 5.20 provides an overview of CADBIT II's Insertion function which assists designers with insertion of BIT technique circuitry into CAD schematics. CADBIT II identifies the required part types and automatically calculates the number of each part type required as a function of CUT size and other parameters.

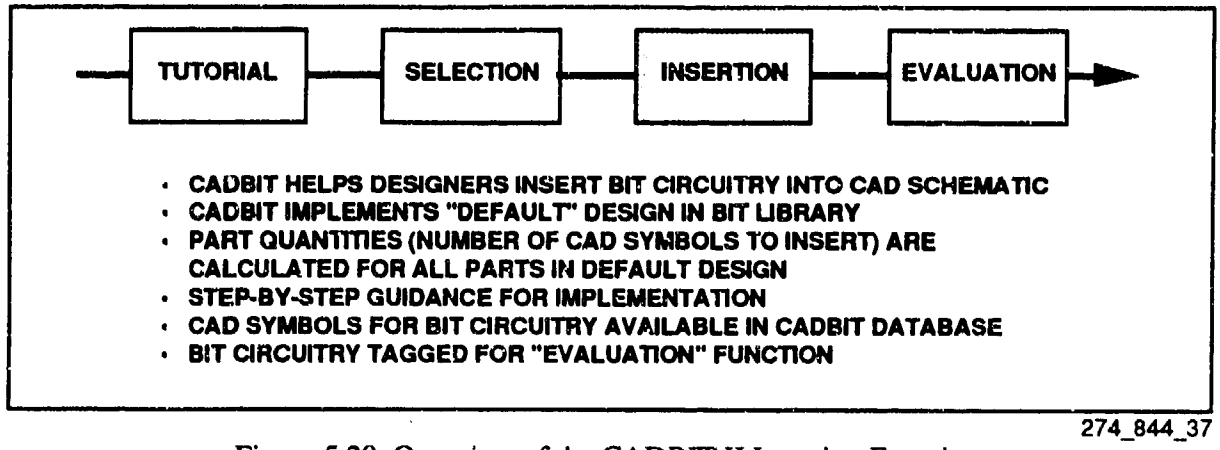


Figure 5.20 Overview of the CADBIT II Insertion Function

Figure 5.21 shows the logical flow of the CADBIT II Insertion process. The user's actions are indicated on top with the associated CADBIT II process steps shown below them. The flowchart shows a loop in which designers implement the BIT technique one subfunction at a time. A BIT "subfunction" is a logical subset of the BIT technique such as those shown in the typical BIT anatomy in Figure 5.3.

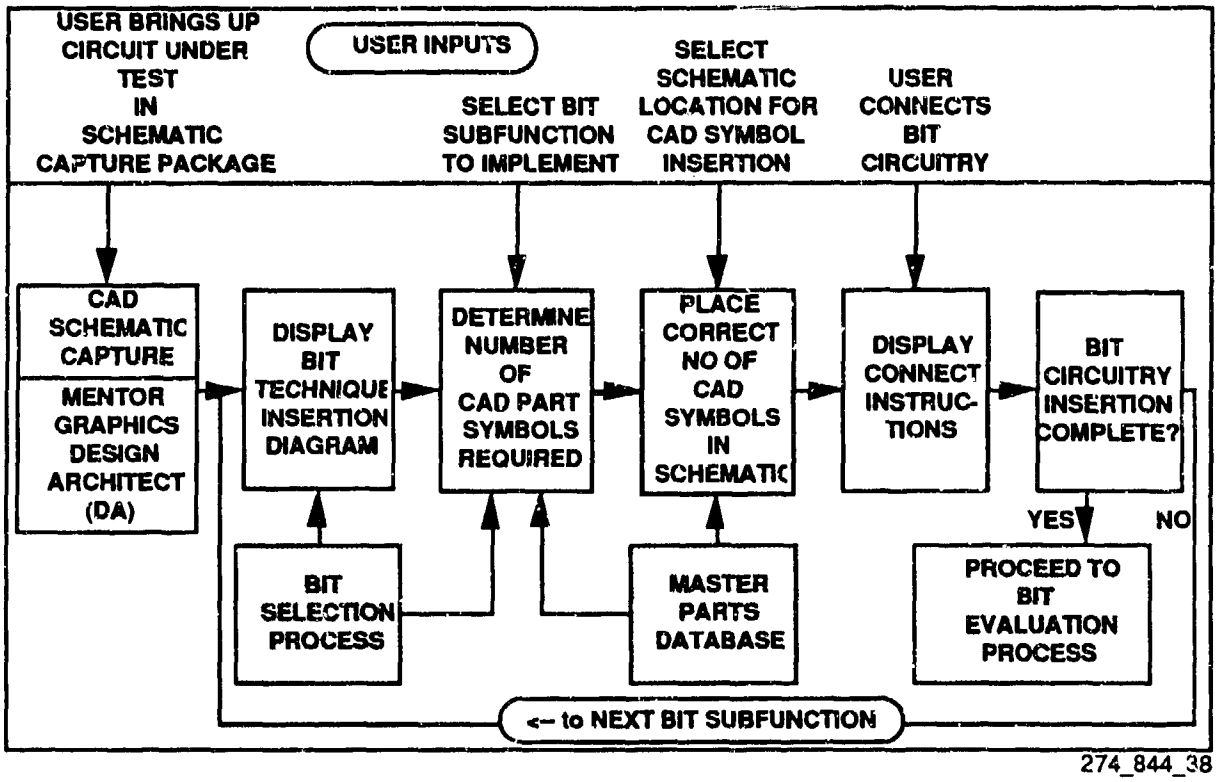


Figure 5.21 CADBIT II BIT Insertion Process Flowchart

DEFAULT DESIGN

As in the Selection function Penalty Equations discussed above, the target design for BIT Insertion is the "Default Design" for the selected BIT technique. CADBIT II stores this Default Design in the BIT Library database in the format of the CAD Schematic Capture software package (Mentor Graphics' Design Architect). Figure 5.22 shows the important features of the Default Design.

The Default Design is not a unique solution to implementing the selected BIT technique. The Default Design serves as a template illustrating the important concepts and issues involved with implementation.

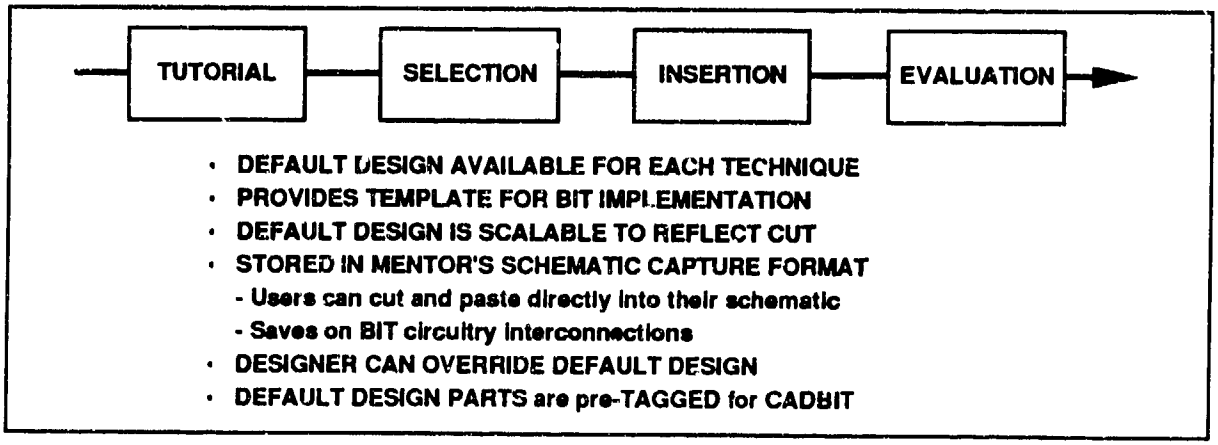


Figure 5.22 The BIT Insertion function implements the "Default Design."

Figure 5.23 shows an example Default Design for the "Voltage Summing" technique. Details of this technique may be found in the second volume of this Final Report. Briefly, this technique sums analog voltages output from a Circuit Under Test and uses a Window Comparator to check if the voltage sum is within an acceptable range. A BIT status "Pass/Fail Flip/Flop" is set to indicate the results.

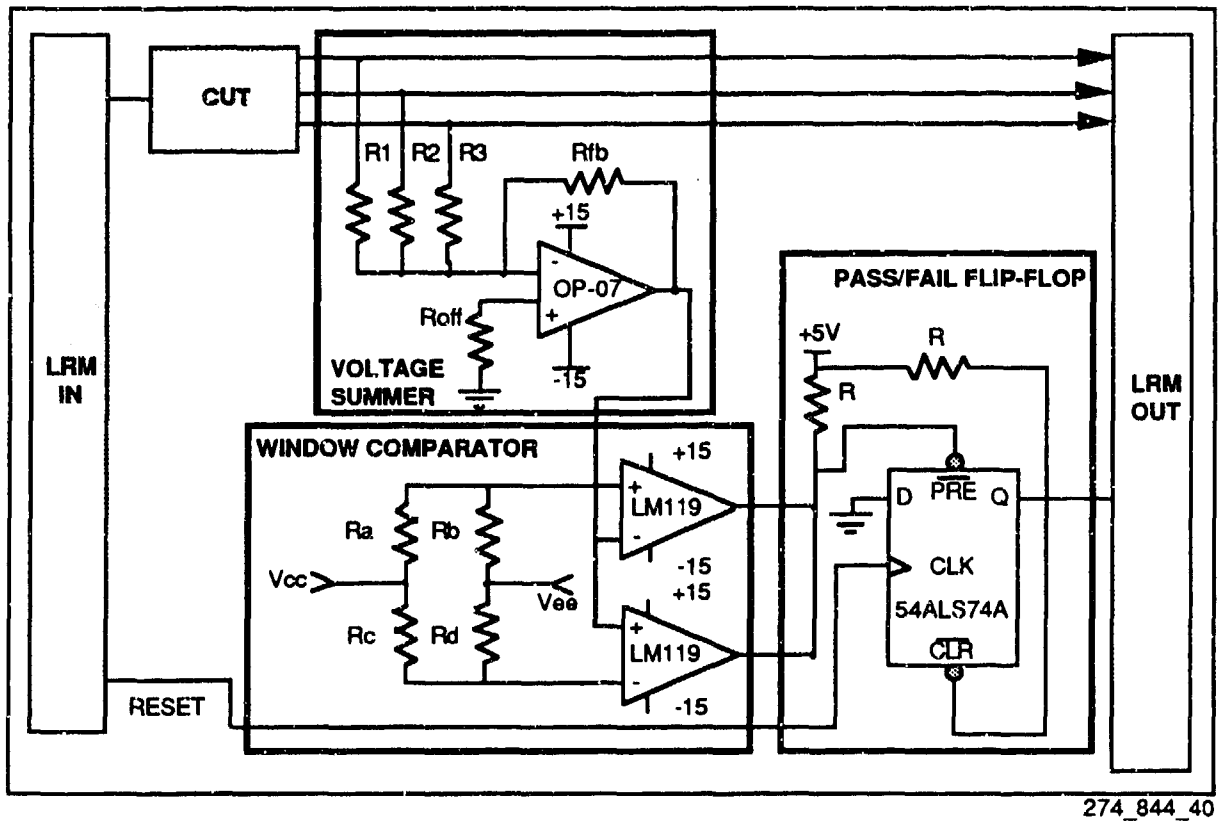


Figure 5.23 Example Default Design for the "Voltage Summing" Technique

BIT TECHNIQUE INSERTION DIAGRAM (BTID)

CADBIT II uses a top-level block diagram, called the BIT Technique Insertion Diagram to guide the designer with step-by-step BIT technique implementation. The BTID contains a series of blocks representing the various subfunctions of the BIT technique that must be implemented. The designer sequentially proceeds with CAD Insertion one block at a time. As the subfunctional blocks are completed by the designer, CADBIT II shades the appropriate boxes in the BTID.

Figure 5.24 shows an example BTID for the "Voltage Summing" BIT technique. This example corresponds to the Default Design shown in Figure 5.23. All components in the Default Design can be mapped to a subfunctional block in the BTID.

To help designers implement a given "block" in the BTID, CADBIT II identifies the parts required (per the Default Design) and calculates (per the Component Determination Equations) the number of each part type. Component packaging equations are not used since the designer is inserting individual CAD symbols for the BIT circuitry. CADBIT II calculates the number of units or CAD symbols to place in the designer's CAD schematic. For example, if a BIT technique uses three inverters, three individual inverter symbols are placed in the designer's schematic. We have seen that CADBIT II's Default Design will use one Hex Inverter package to implement these three inverters.

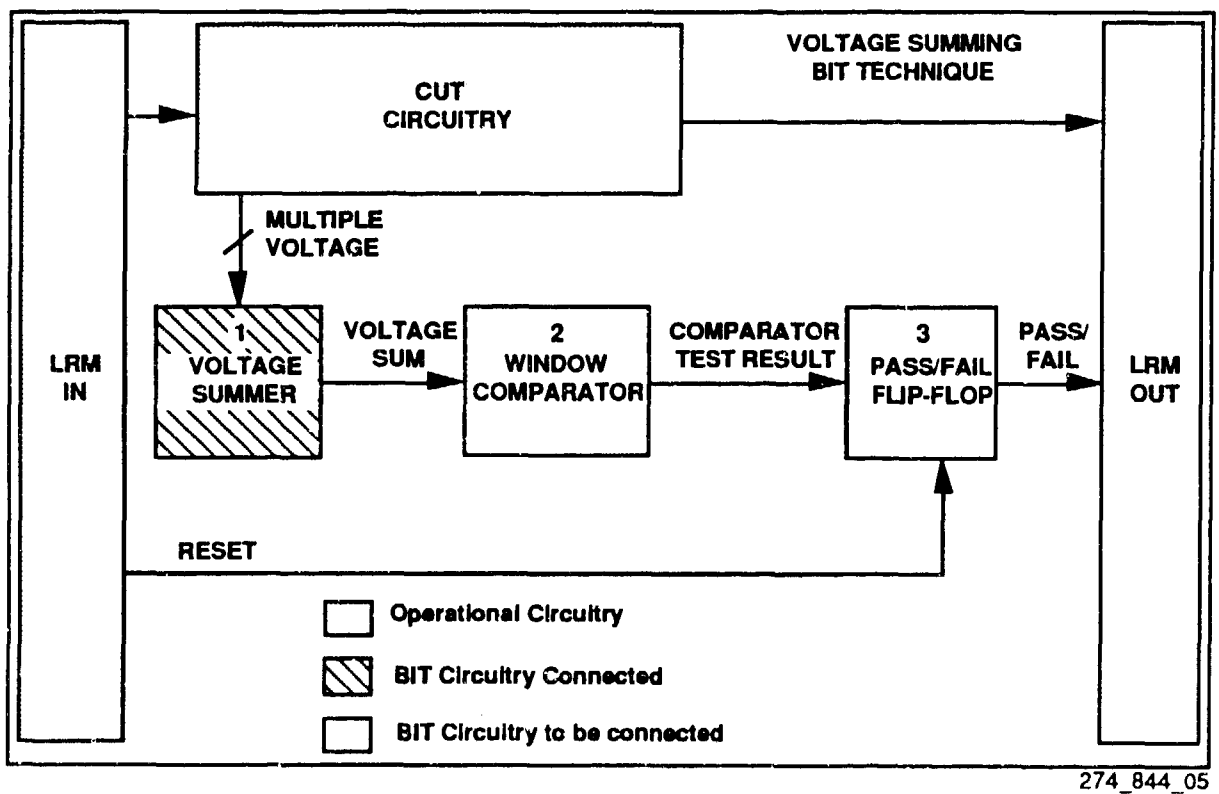


Figure 5.24 BTIDs Guide the Designer with Step-by-Step Implementation

CONNECTION

The designer selects the "sheet" of the CAD schematic in which CADBIT II places the parts required to implement a BIT subfunction. CAD symbols for the BIT components will appear above the designer's current schematic circuitry. CADBIT II then assists the designer with step-by-step connection instructions. "Power" users can take advantage of the Default Design which is pre-connected and available in the format of the CAD Schematic Capture software package. By cutting and pasting from this Default Design, the user can greatly speed up the connection process, especially using the existing interconnections of BIT elements inherited from the Default Design.

Designers can override the Default Design and implement the BIT technique using other parts and other technology families. To enable subsequent impact analysis by the CADBIT II Evaluation function, all parts used must be registered in the Master Parts Database (See the section on the BIT Library Encoding Utility).

CAD symbols for all BIT circuitry used by CADBIT II are contained in the BIT Library Master Parts Database so that no commercial CAD part libraries are required to exercise the CADBIT software. Since CAD symbols in the Master Parts Database do not contain part modeling information, designers should use part symbols from an appropriate commercial CAD part library if they intend to follow BIT Insertion with circuit simulation. CADBIT II "tags" each part used for BIT with an identifier so that the Evaluation function can recognize which components in the designer's schematic were used for BIT and which were used for normal operational circuitry. Designers must similarly "tag" parts taken from other part libraries to enable use of the Evaluation function.

5.6 EVALUATION

Figure 5.25 provides an overview of CADBIT II's Evaluation function. CADBIT II evaluates the impact of added BIT circuitry on three important design parameters: board area, power, and weight.

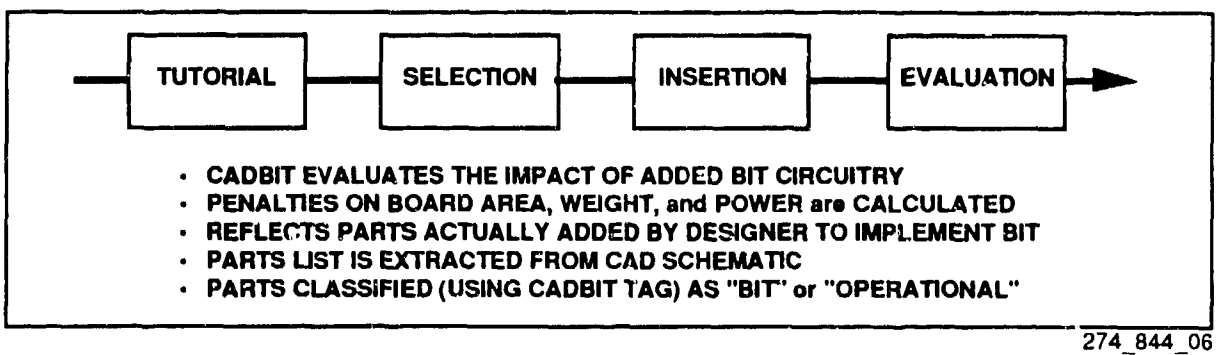


Figure 5.25 Overview of the CADBIT II Evaluation Function

CADBIT II uses Penalty Equations similar to those in the Selection function. We saw that a typical Area Penalty Equation looked like:

Area Penalty = Area of components and traces to implement BIT
= $n_1*a_1 + n_2*a_2 + ...+ n_i*a_i + ...$ + 15% for traces

Where:

n_i = Number of i-th part type required to implement BIT
a_i = Area of i-th part type

There are several differences between the Penalty Equation calculations performed by the Selection and Evaluation functions. The Selection function calculates the Penalty Equations associated with the Default Design. The Evaluation function is performed on the actual design as implemented by the designer, This design may differ from the Default Design for many reasons. For example, the designer may implement the design using a different technology family or take advantage of normal operational circuitry already in the CAD schematic that can be used for BIT implementation.

Whereas the Selection function uses CDEs to calculate the number of parts ("n_i"s) as a function of underlying CUT variables, the Evaluation function extracts all parts from the designer's schematic and directly counts the number of parts used to implement the BIT technique. Component packaging equations are applied to convert the number of units found into an equivalent number of physical component packages. A "CADBIT tag" appended as a property to the CAD symbol is used to identify BIT components and separate them from normal operational circuitry.

As before, the areas of the components ("a_i"s) come from the Master Parts Database. This is why parts used by the designer which override the Default Design must be registered with the CADBIT II Master Parts Database.

A BIT Technique Penalty Report provides part-by-part area, weight, and power penalties and is summed to give total "board" impact of implemented BIT techniques. Multiple BIT techniques per "board" are recognized by the Evaluation function software. Penalties are reported individually for each BIT technique.

Figure 5.26 provides a flowchart of the Evaluation function. This function tends to be the most "push-button" of the basic designer functions. The user merely brings up the CAD schematic containing the CUT and completed BIT implementation, selects the portion of the schematic to evaluate, and then analyzes the results of CADBIT II's evaluation.

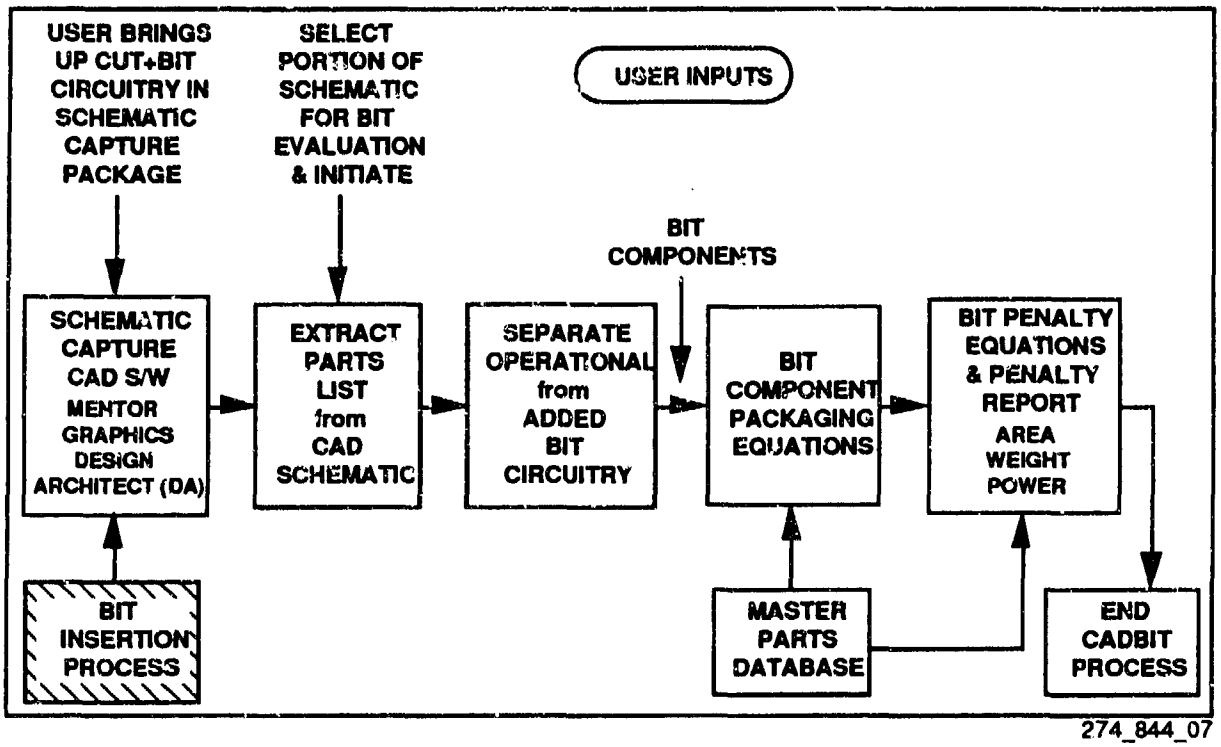


Figure 5.26 Evaluation function measures impact of BIT implementation.

5.7 OTHER SYSTEM FUNCTIONS

In addition to the functionality discussed in Section 5.1 to 5.7, the CADBIT II System has three additional capabilities. These capabilities are Encode BIT Library, on-line help, and VHDL file input.

ENCODE BIT LIBRARY

The power and flexibility of the CADBIT II System is derived from the BIT Library. It is envisioned that many additions and enhancements to the BIT library will occur after the system was delivered. For this reason, a utility is provided to add, modify or delete BIT library data. This utility is the Encode BIT Library functionality of CADBIT II.

Multiple text and graphics files comprise each BIT technique. The process from the CADBIT I specification was simplified by providing a man-machine interface with narrative and prompting menus. The Software User Manual addresses the Encode BIT Library utility, detailing the composition of the library database.

ON-LINE HELP

Help is provided by three mechanisms in the CADBIT II System. The tutorial data (as explained in Section 5.4) provides detailed data on each technique in the BIT Library. Process dependant help is available through the use of a '?' in any text field or the selection of the help button in a window. Upon entering a '?' or selecting the help button, a window is displayed providing the user with help on the requested topic.

VHDL FILE INPUT

Mentor Graphics Idea Station supports VHDL file input via their 1076 VHDL compiler. Mentor Graphics provides a VHDL compiler for developing, interactively debugging, and simulating design. VHDL provides an Ada-like language for describing the function of electronic design. VHDL file input into Mentor Graphics was tested during Acceptance Testing.

6.0 CADBIT II SOFTWARE

The CADBIT II software was written using industry standards in C, Unix, X Windows and OSF/Motif. The system development approach (Figure 6.0) took these standards and the software design to program the software module. The results of the BIT technique review aided the design of the Encode BIT Library utility. The designed module was integrated with Mentor Graphics Design Architect software package, tested and delivered to Rome Laboratory. This section describes the design process, integration with the CAD software, and transportability issues. A structured analysis approach (Figure 6.1) was used to yield a highly modular and maintainable software module.

6.1 SOFTWARE REQUIREMENTS

The purpose of requirements analysis is to define the job to be done and the acceptable criteria consistent with the constraints, cost schedule and resources. The CADBIT I Software Specification was reviewed and redlined to separate the document into requirements, design specifications and user manual data. The requirements were taken and merged with additional requirements from the CADBIT II Proposal and Statement of Work to generate the CADBIT II Software Requirements Specification (SRS). The requirements that were detailed in the SRS were used to generate of the CADBIT II Software Test Plan. Table 6.0 lists the key software requirements.

The unobtrusive requirement deals with the interaction of the CADBIT II software module and the engineer's design process. CADBIT II was written in a manner that does not interrupt the design process. Section 6.8 discusses the transportability requirement. All documentation was written to 2167A standards and the structured analysis process was used to develop a modular, highly portable, easy to maintain software module. Section 5.8 discusses the on-line help available and configuration control was implemented using the SCCS (Source Code Control System) utility. SCCS is explained in more detail in the attachment.

6.2 SOFTWARE DESIGN

The design data from the CADBIT I Software Specification was reviewed to determine where improvements and simplification could be made in the CADBIT II implementation. A regular set of peer reviews were used throughout the preliminary design phase to evaluate the software architecture. The results of these reviews were a clearer understanding of the CADBIT II requirements and the need for a simple user interface design.

TABLE 6.0 KEY SOFTWARE REQUIREMENTS

Key Software Requirements
Shall operate in an unobtrusive manner
Shall be transportable across a variety of CAD workstations, portability will be maximized
Shall be developed using 2167A standards
Shall be modular and highly portable
Shall be easy to maintain and enhance source code
Shall be placed in configuration control
Shall have extensive on-line help
Shall adhere to C and Unix standards

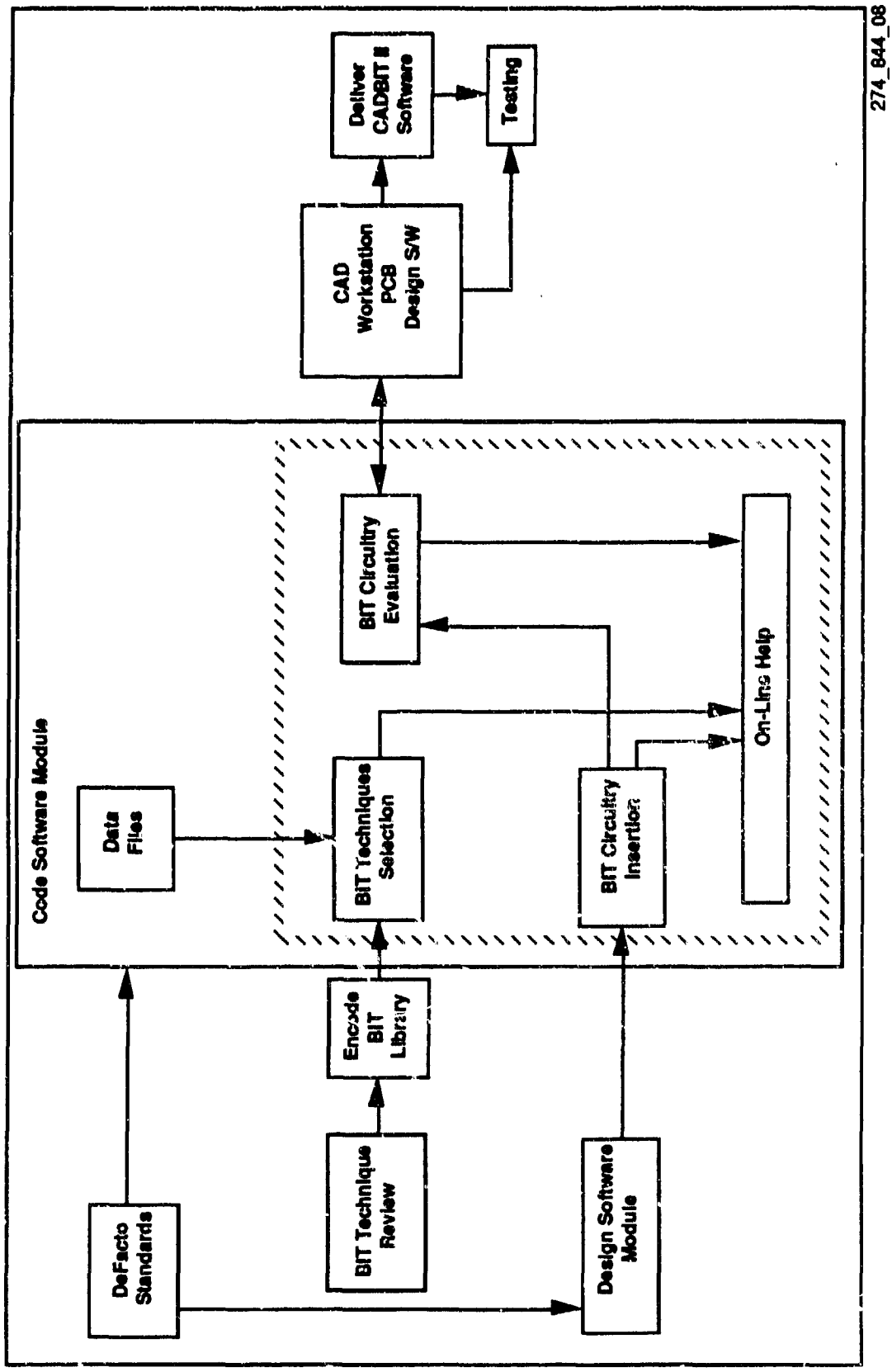


Figure 6.0 CADBIT II System Development Approach

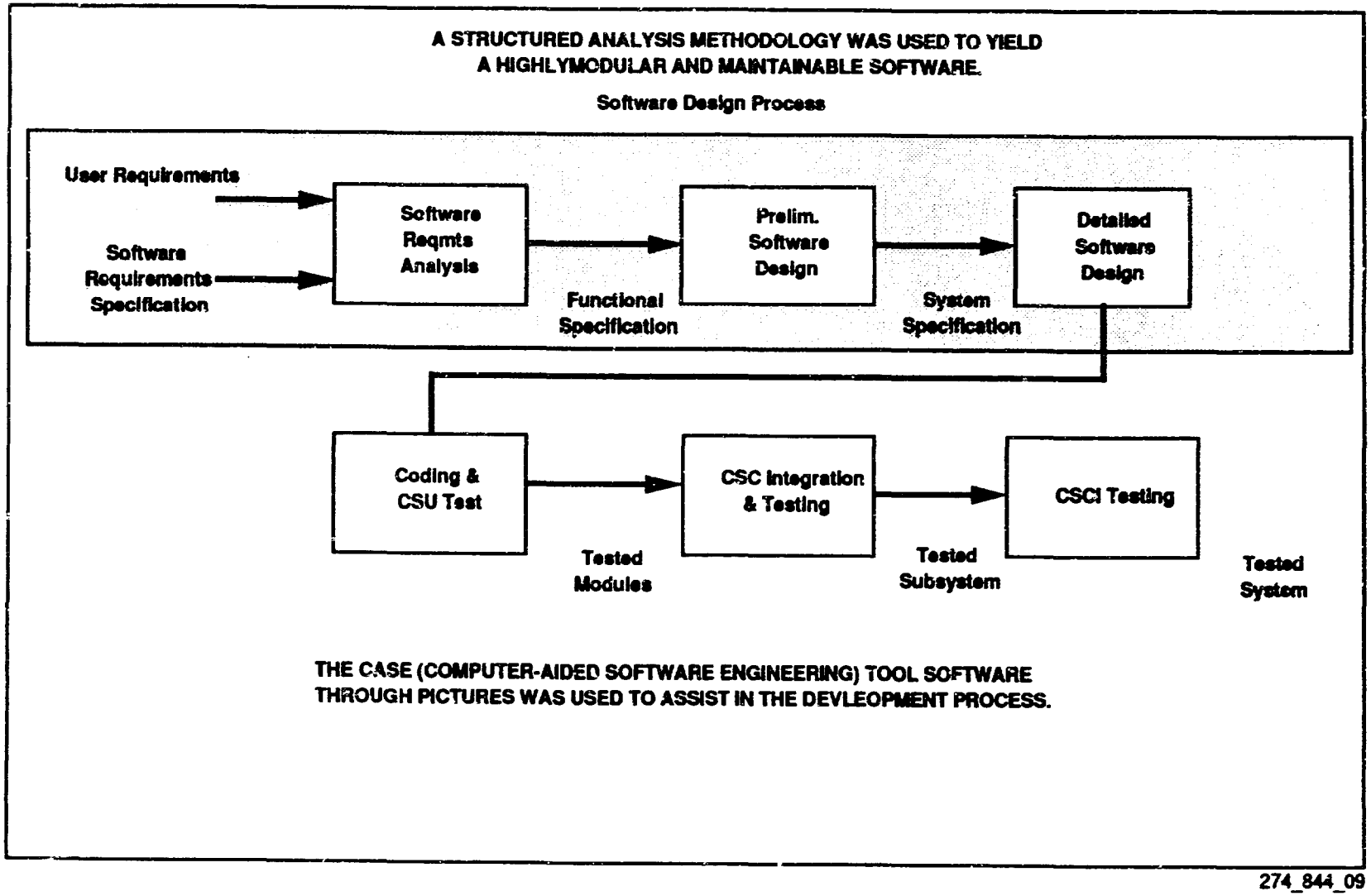


Figure 6.1 CADBIT II Software Development Process

6.3 SOFTWARE DEVELOPMENT

The CADBIT II software was programmed in C and developed in three incremental builds or phases. Each phase extended the previous phase in terms of incorporating additional design information and functionality. The content and scheduling of each phase is shown in Figure 6.2. Phase 1 implemented the initial requirements and preliminary design and produced a working and testable shell. This shell contained the main menu interface, the BIT techniques text files, and an example of a complete tutorial. Phase 2 filled in this software shell with the capability to perform BIT selection, BIT evaluation and access to the encoded BIT library. Finally, Phase 3 contained all required functionality including BIT insertion, help screens and a library encode utility program. This final phase produced end-product code which was fully tested and was validated during the acceptance test programs. The approach to CADBIT II software development was documented in the Software Development Plan that, along with program design standards, facilitated ease of changes and/or additions to CADBIT II.

6.4 COMPUTER STANDARDS

The CADBIT II software module conforms to the de facto standards of C programming language and Unix operating system. By adhering to standard C function calls, the portability between differing CAD workstations was enhanced. The CADBIT II software design accounts for variations in Unix versions therefore making adaptation a straightforward process. Section 6.8 discusses transportability in more detail.

6.5 ELEMENTS AND DESCRIPTION

As depicted in Figure 6.0, the CADBIT II software is composed of the following elements:

BIT Technique Selection
BIT Circuitry Insertion
BIT Circuitry Evaluation
Integration with Design Architect
Encode BIT Library
On-line help
Data files

Each element listed above interacts with user inputs and data files. BIT Technique Selection aids in the selection of appropriate BIT techniques for the circuit under test. CADBIT II compares the suitability attributes lists of each BIT technique with the attribute list of the circuit under test. This comparison yields a list of suitable BIT techniques. These suitable techniques are ranked using component determination equations and penalty equations. Section 5.5 and the Software User Manual describe the selection process in more detail.

BIT Circuitry Insertion guides the user in inserting the desired BIT into the circuit under test. CADBIT II will select the necessary components from the CADBIT parts library and place them into Design Architect. The user is then instructed on how to connect the BIT circuitry. Section 5.6 and the Software User Manual describe the insertion process in more detail. BIT Circuitry Evaluation provides a report of each BIT component used with the circuit under test. CADBIT II extracts actual data from the schematic in Design Architect and calculates the added area, power and weight the BIT circuitry added. Section 5.7 and the Software User Manual describes the evaluation process in more detail.

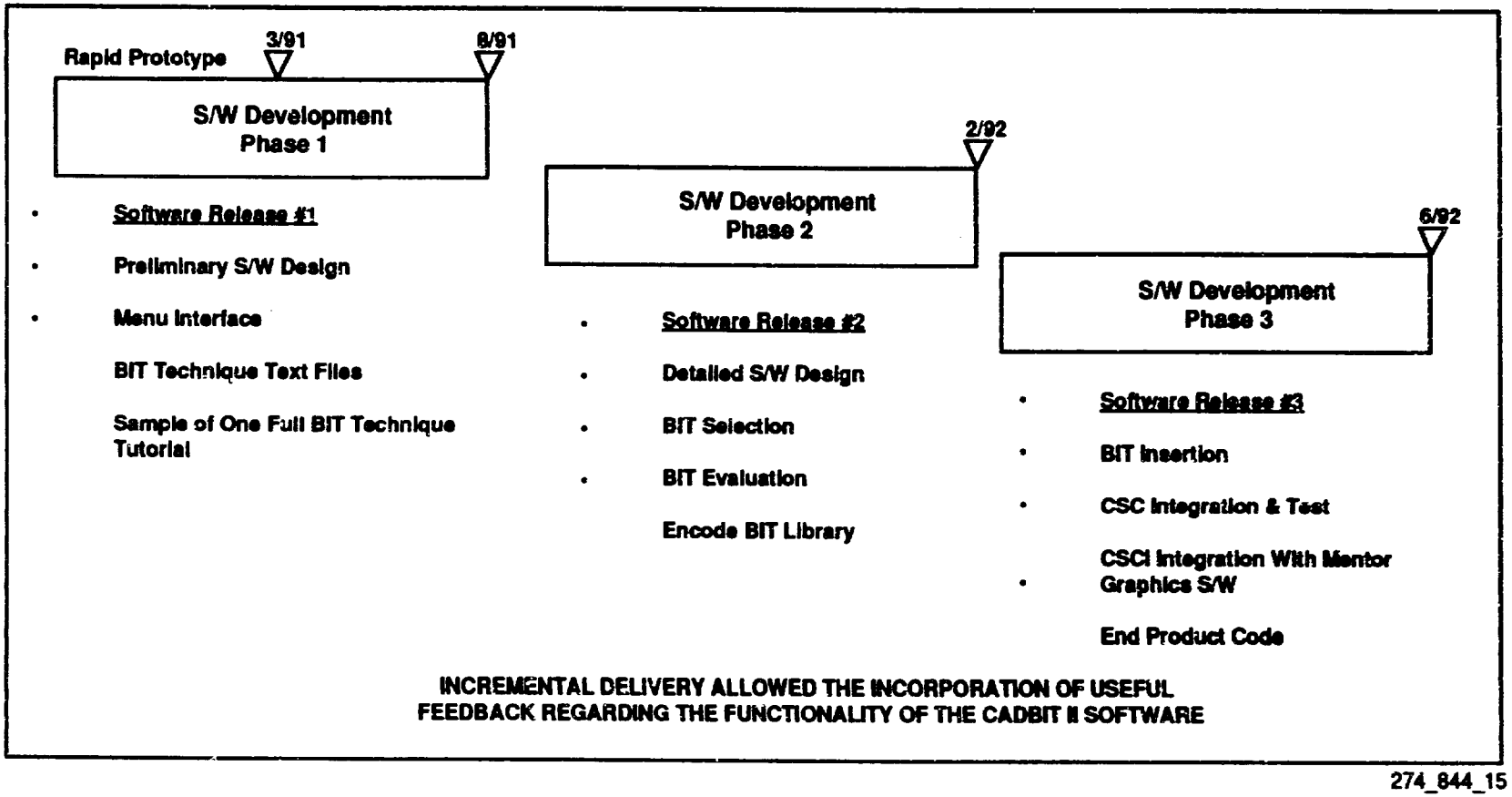


Figure 6.2 CADBIT II Incremental Software Development Process

The CADBIT II software module interacts with Design Architect during tutorial, insertion, evaluation and Encode BIT Library. CADBIT II communicates with Design Architect via an inter-processor communication port. This communication is described in Section 6.7. The Encode BIT Library function allows the addition, modification or deletion of BIT techniques from the BIT library. Each BIT technique is composed of textual, graphical and data files. The Encode BIT Library function allows the user to modify all elements of the BIT library. Section 5.8.1, the Software User Manual and the System Administration Information Attachment give more information on the Encode BIT Library functionality.

On-line help is achieved through the retrieval of specified text files. The data for each help topic is stored as a text file and displayed by the software when needed. All help files are located in one sub-directory.

There are numerous data files that interact with the CADBIT II software module. Data files can be grouped into the following four categories:

BIT Library Data. The BIT library data is composed of text files, graphics files and data files. Table 6.1 lists the data necessary to support the BIT library.

TABLE 6.1 BIT LIBRARY DATA FILES

Master question list
Master parts data table
Advantages of each BIT
Disadvantages of each BIT
Bibliography of each BIT
BIT Technique Insertion Diagram for each BIT
Flow chart description for each BIT
Default design of each BIT
Block diagrams for each BIT
Parts data table for each BIT
Suitability Attribute List for each BIT
User Requested data for each BIT
List of all BITs in library
Library of parts

Profile Data. Profile data files contain data regarding each Circuit Under Test that has been processed by CADBIT II. Information stored for each circuit under test (profile) is listed in Table 6.2.

TABLE 6.2 PROFILE DATA FILES

List of all profiles
Bit selected for each circuit under test
BIT Technique Penalty report for each circuit under test
Insertion status for each circuit under test
Results of component determination equations
List of suitable BITs for each circuit under test
List of rejected BITs for each circuit under test
User Design Profile for each circuit under test
Answers to User Requested Data for each circuit under test

CAD Communication Data. There are several files that CADBIT II uses to communicate with Design Architect and this process is explained in section 6.7.

Help Files. Help files were discussed earlier in this section.

6.6 USER INTERFACE

One area recommended for re-examination of the CADBIT I specification was the methodology used for interacting with the user and the CAD software. Due to advances in technology since CADBIT I, more efficient, user-friendly methods for implementing CADBIT II were available. The X Window standards was chosen as an alternative to using C-shells for interface creation and custom menus within the CAD system. Custom menus within a CAD system are time-consuming to create, nontransportable across differing CAD workstations and PCB design software packages and complicate the user interface.

The user interface was considered as an integral design requirement during software development. The goal was to insure that the user interface was simple and intuitive. This meant that the screen layouts, control and navigation methods selected needed to be logical to the end user. On-line help was considered a basic ingredient to this design concept as was the ability of the user to navigate the system without fear of being lost inside the CADBIT II process.

The CADBIT II user interface was built using the X Window System. The X Window System is a software platform for building windowing systems. The X Window System was developed by MIT and is controlled by an X consortium. The X Window System is the de facto industry standard which allows windowing applications to be portable across a class of machines.

The user interface was written using OSF/Motif (or Motif). Motif is a graphical user environment based on a published style guide. Motif is an interface toolkit with a widget set. A widget is "a collection of an X window and its associated input and display semantics–which is dynamically allocated and contains state information. Widgets are smart windows. They handle many of the housekeeping chores of X internally without application intervention. Windows replace the big event-based switch with a list of instructions to "call me back when a certain event occurs.

Once the basic software design was available, it was determined that a prototyped version of the user interface was necessary to evaluate the viability of the proposed design. This prototype was developed using X windows and Motif icon/widgets. A simplified scenario was prepared so that the basic prototype could be reviewed and critiqued by electronic designers. The feedback received from these users provided the CADBIT team with significant insight into areas of improvement related to the user interface and software features. The CADBIT II prototype duplicated the "look and feel" of this type of user interface so that user evaluations could be conducted early in the software design process. This timely feedback provided the CADBIT II design team with valuable insights into the way the user community would employ an engineering tool of this type during the design of PCB circuits.

The CADBIT II user interface and its operations are described in the Software User Manual (SUM).

6.7 INTEGRATION WITH CAD SOFTWARE

The CADBIT II software interfaces with Design Architect for displaying CAD graphical files, inserting BIT circuitry and performing data extraction during Evaluation or Encode BIT Library. CADBIT II communicates with Design Architect using an ipc (inter-processor communication) port available in version 8 of the CAD software. By use of a startup file, Design Architect looks for commands in a file called da_test. This file is created in the user's home directory by the CADBIT II software. Once this file is created, Design Architect processes the commands in the file. After creation of the da_test file, CADBIT II waits for Design Architect to process the commands. Design Architect signals completion of processing by creation of a file called da_done in the user's home directory. Once this file is created, CADBIT II continues processing.

The commands written out to the file da.test are commands to invoke Design Architect intrinsic functions or userware. The intrinsic functions are used to open files while userware was written to perform data extraction and instantiation of parts. The userware is written in AMPLE (Advanced Multi-Purpose Language). AMPLE, a part of Mentor Graphics version 8, is an interactive, compiled, end-user programming language. AMPLE is a powerful user-extension language supporting the need to modify and extend the functionality of the Mentor Graphics software.

The CAD insertion function "instantiates" each component required to implement the selected BIT technique. Instantiation refers to the placement of representative CAD symbols for each required BIT component into the designer's schematic. To instantiate the parts required to implement a BIT technique, CADBIT II creates two data files. One file contains the parts to instantiate and the second file (named da_test) contains the commands to run the necessary userware. The userware to instantiate the parts will place the new parts above the current schematic. This isolates the BIT circuitry and PCB design so the user can clearly see what parts were instantiated. All userware is defined in the Mentor Graphics startup file and is loaded when Design Architect is invoked. The Software User Manual details how to set up this startup file.

6.8 TRANSPORTABILITY

The major issues that must be addressed when examining software transportability are:

OPERATING SYSTEM

To maximize the ease with which the CADBIT II software can be transported between differing engineering workstations, a common version of the Unix operating system was used. The CADBIT II software was designed and coded in an manner that maximizes transportability. This was accomplished through the use of a software interface layer between CADBIT II and the operating system as described in the Software Design section below.

PROGRAMMING LANGUAGE

Adherence to C programming standards during software design and development ensures maximum transportability and compatibility between systems. Many C compilers contain additions to the standard as defined by Kernighan and Ritchie making programs using these additions machine dependant. By adhering to the defined standard, we maximized the probability that our software would be transportable to different workstations without major modification.

SOFTWARE DESIGN

Regardless of the operating system and language chosen for software development, variations between computer systems do not allow most programs to be directly transportable. For example, a call to the common *get* command on a Silicon Graphics workstation must be called using *sccsget* on a Sun workstation even though both machines use the Unix operating system. To maintain the ease with which software can be transported between machines, we have used a technique called layering.

Layering allows us to isolate machine- or application-dependent program statements and commands. These statements and commands usually involve interaction with the operating system or another software application. We identified all machine and application-dependent calls and placed each one in its own C function. Each C function was given a descriptive name and contains only a call to the machine or application-dependent routine. These functions were layered so that when transporting to another system, usually only one line of code in our software layer will have to change. Figure 6.3 illustrates the concept of layering.

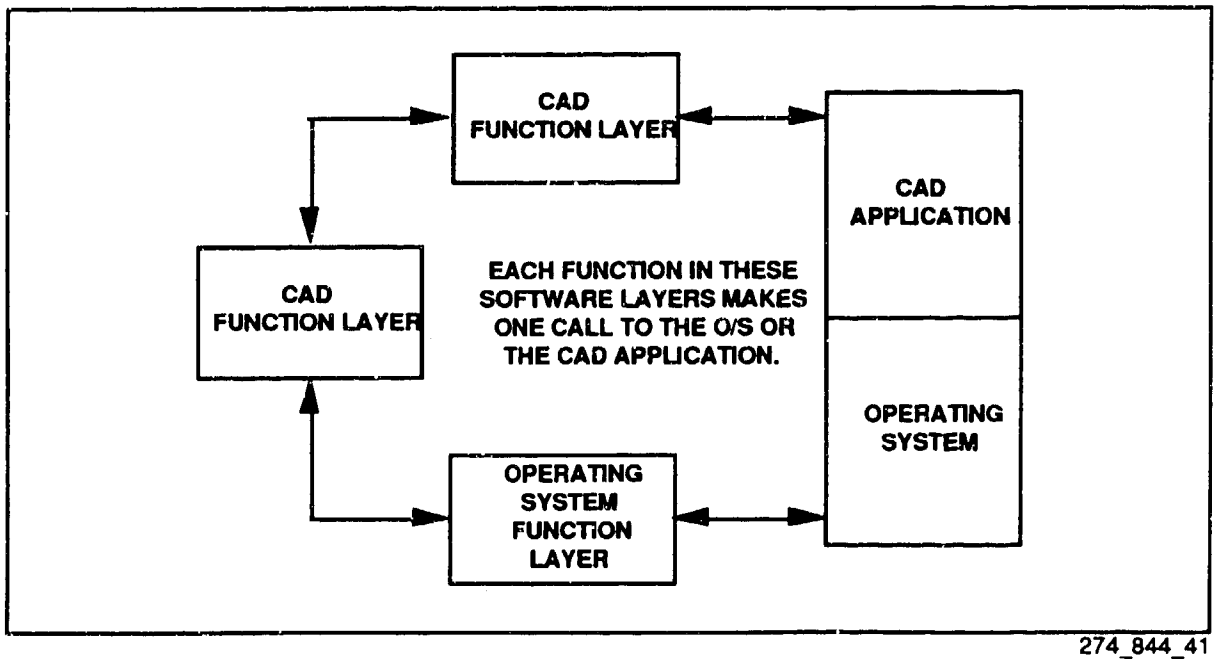


Figure 6.3 CADBIT II Layering Concept

SOFTWARE CODING STANDARDS

We drew from extensive internal software development standards for CADBIT II design and coding. For example, our use of standard headers in all programming routines increases the ease with which the software can be modified and maintained. Ease of maintainability and modifiability are especially important when BIT techniques are to be added or deleted. Design Coding Standards from Appendix B of DOD-STD-2167A appear in Table 6.3. These standards were supplemented by internal software development standards where deemed necessary.

TABLE 6.3 ITEMS IN CODING STANDARDS

Indentation
Use of capitalization
Uniform presentation of information throughout the source code
Use of headers
Layout of source code listings
Conditions under which comments are provided and the format to be used
Size of code aggregates

WINDOW MANAGER

Use of a standard window manager, such as X Windows and Open Systems Foundation (OSF) Motif, greatly enhances portability of the CADBIT II software.

7.0 SYSTEM TEST

The HMSC testing approach was based on a structured process for implementing both informal and formal tests. This process, shown in Figure 7.0, is in full compliance with HMSC Software Development Practices and Procedures and the tailored requirements from DOD-STD-2167A. The entire test program is described in the Software Test Plan.

7.1 TEST PROGRAM

The Software Test Plan (STP) evolved as the software design matured. The STP was written and updated as the Software Requirements Specification and Software Design Documents matured. The STP was used as the basis for all software testing either formal or informal. The STP identifies each test and lists all test assumptions and constraints. There were three types of verification used for testing. They are:

Visual Inspection: Examination of program listings, logic flows and prototyped screen layouts

Data Analysis: Review of test or analytical data generated to evaluate a specific function.

Demonstration Test: Verification of function by the conduct of predefined tests and the comparisons of test results to specified criteria.

Figure 7.1 shows the software test flow used for the CADBIT II System. Programmer module tests included informal testing, unit testings, component testing and system integration testing. Unit testing involved exercising executable statements and checking for error detection and recovery. Component testing focused on testing the internal interfaces that occur. "Beta" tests were performed by HMSC to validate baseline requirements. System integration testing was designed to bring together the hardware and software to operate in a predefined environment.

For CADBIT II, two formal test were performed. Preliminary Acceptance Testing was conducted at HMSC and all tests in the STP were performed. Final Acceptance Testing was performed at RL using a subset of the tests in the STP. Both Preliminary and Final Acceptance Testing demonstrated the entire range of CADBIT II System capabilities. At the conclusion of Final Acceptance Testing, the software was delivered and demonstrated at RL.

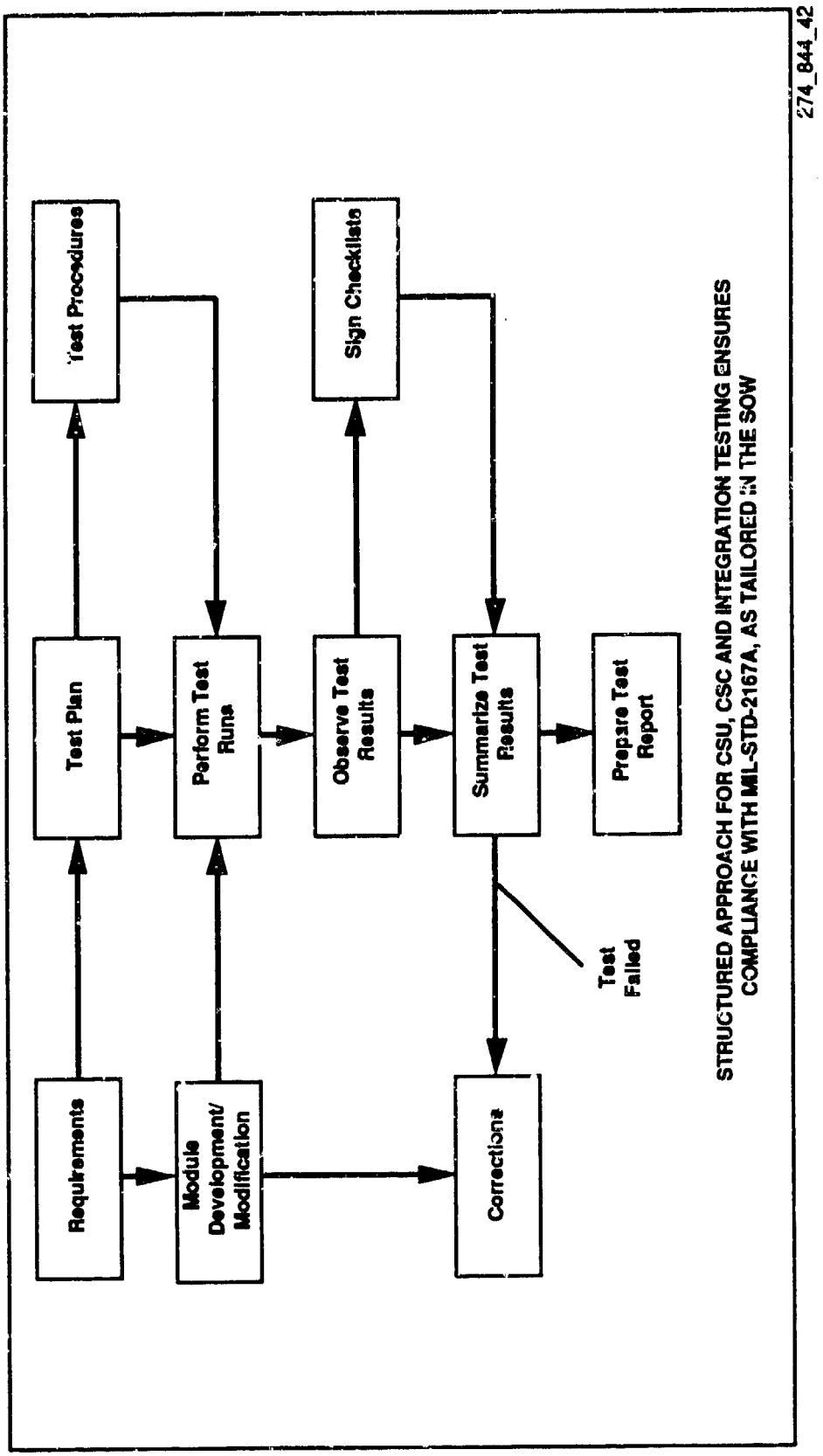


Figure 7.0 Test Approach

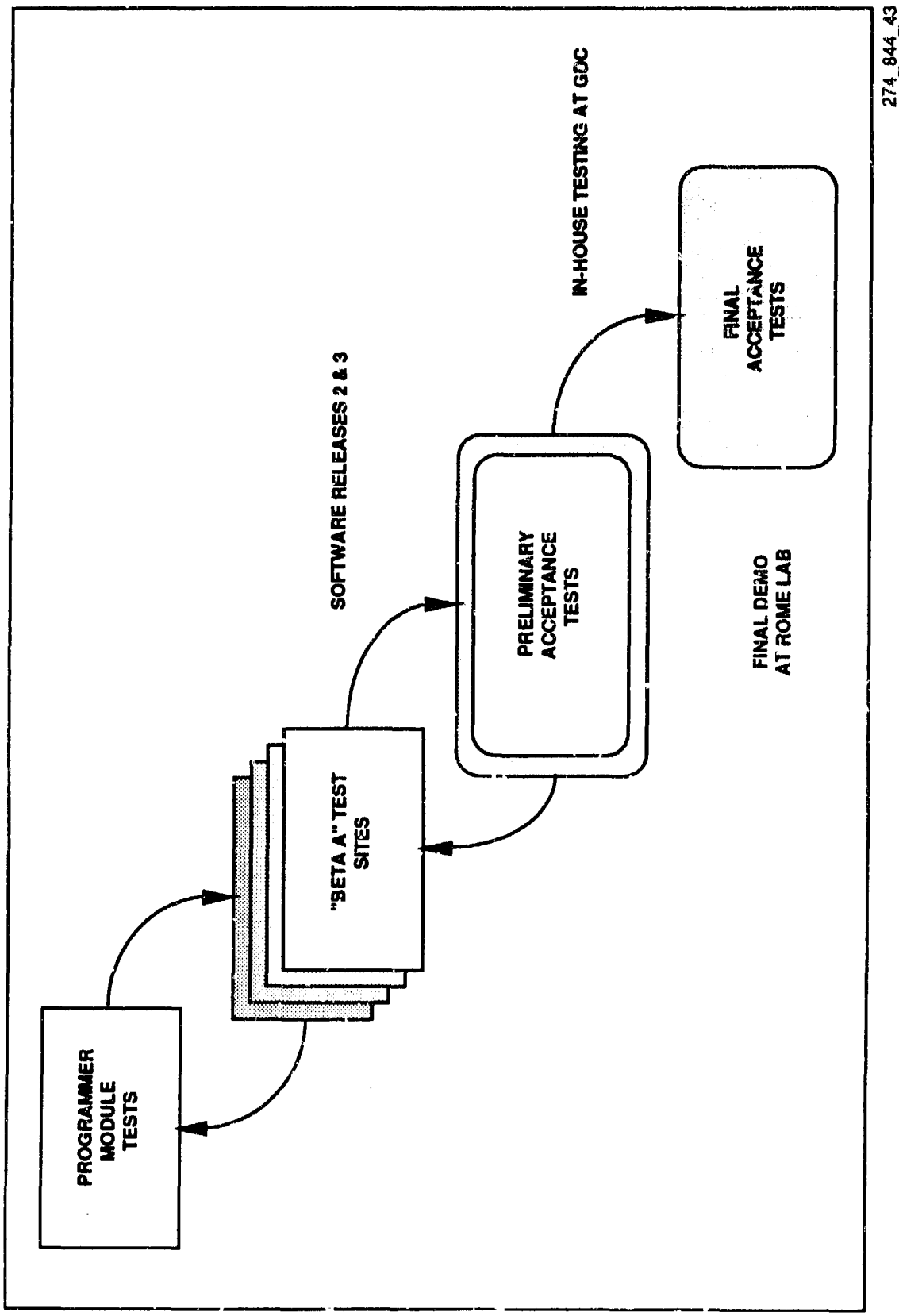


Figure 7.1 Software Test Flow

7.2 PRELIMINARY ACCEPTANCE TESTING

Figure 7.2 illustrates the events that led up to and included Preliminary Acceptance Testing (PAT). The CADBIT II software was developed in three incremental builds or phases as shown in Figure 6.2. In June of 1992, after the release of the final software build (named Build 3), the software module was integrated with the workstation hardware, software and operating system. Real-world circuits were used and all CADBIT II operations were exercised and validated against performance measures defined in the STP. All identified problems were documented by Software Problem/Change Request (SPCR) forms and required software modifications were authorized by an approved change request as documented in the Software Development Plan. Once the CADBIT II System met all specified requirements, formal PAT was conducted at the HMSC facility in San Diego. This formal testing took place from September 21, 1992 to October 10, 1992. As anomalies were noted and logged (per the STP), modifications were made to the CADBIT II System and regression testing was performed. PAT was successfully completed with 13 anomalies reported and resolved. The anomalies fell into the following categories with occurrences in parenthesis.

Discrepancy between BIT equations in software and BIT technique review (3)
Inadequate help screen (5)
User interface functionality (3)
Mentor Graphics configuration (1)
CSC Integration (1)

7.3 FINAL ACCEPTANCE TESTING

Final Acceptance Testing (FAT) was conducted at the CASTLE lab in Rome Laboratory (RL) from October 18, 1992 to October 23, 1992. Prior to formal testing, the workstation in CASTLE was configured for the proper operating system, X Window release and Motif version. The Mentor Graphics software was previously installed and its integration with CADBIT II was verified. A subset of the tests in the Software Test Plan (STP) were performed during FAT. This subset was documented in the approved STP. Testing ranged from installation of the CADBIT II System to exercising and validating all functionalities of the CADBIT II System. Eight anomalies were identified during FAT. These anomalies were categorized as follows (occurrences appear in parenthesis):

Array dimension errors (6)
VHDL (1)
Motif Initialization (1)

The array dimension errors were not discovered during PAT due to PAT regression testing being only partial regression testing. The VHDL anomaly dealt with the interaction of Design Architect with the operating system. The operating system in CASTLE was a more current release than that of the baseline noted in the STP. With assistance with Mentor Graphics, the problem was resolved by modifying the test procedure to use the VHDL compiler from the command line not within Design Architect. The Motif initialization error occurred due to improper sequence of calls. This sequence was corrected. After all anomalies were resolved, complete FAT regression testing was performed on the CADBIT II System.

A CADBIT II System demonstration was held concurrently with FAT and the final oral presentation. This demonstration took a sample circuit through the selection, insertion and evaluation functions of the CADBIT II System. The demonstration was given to approximately 20 engineers at Rome Laboratory.

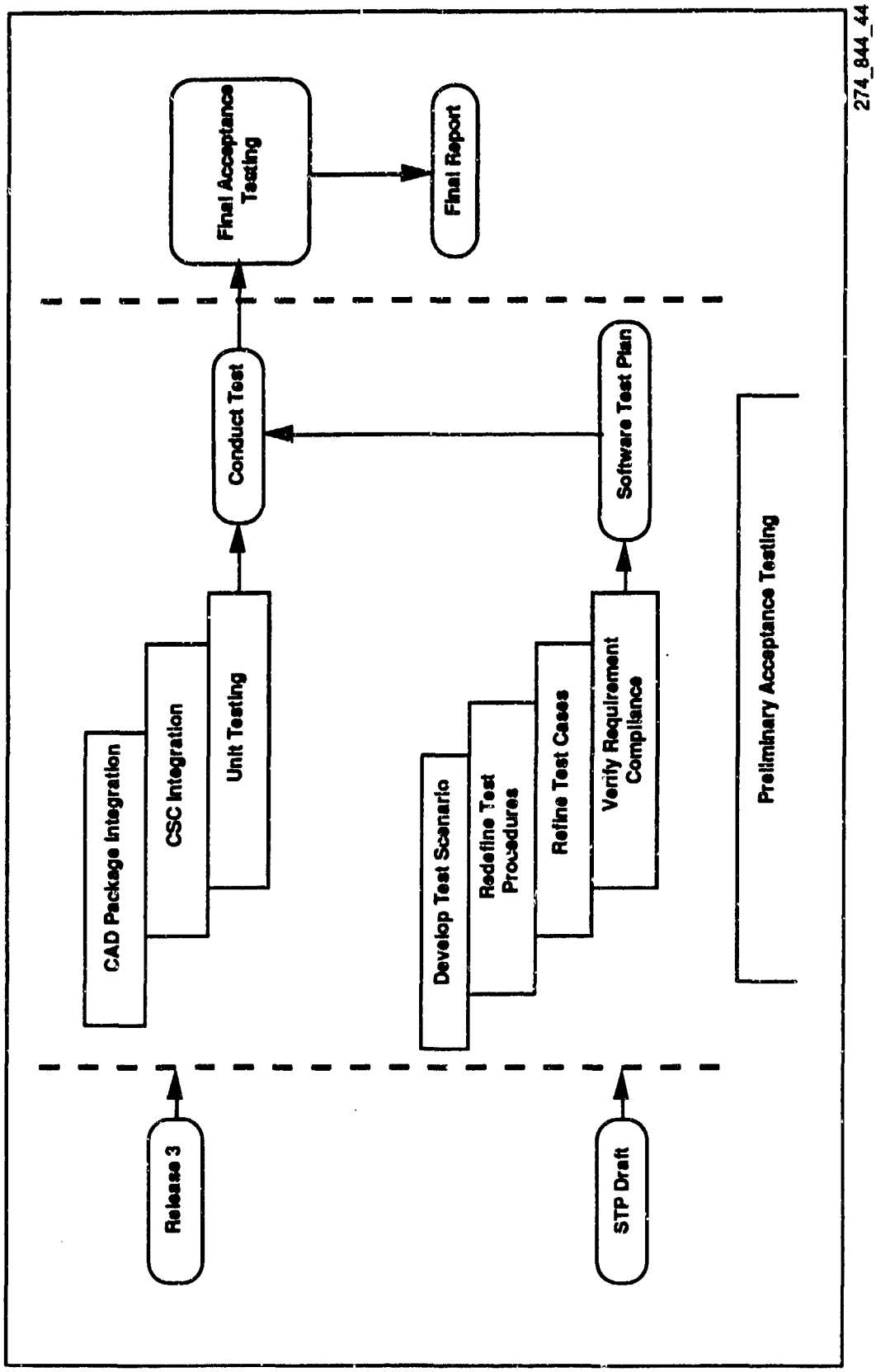


Figure 7.2 Preliminary Acceptance Testing

8.0 PRINTED CIRCUIT BOARD TESTBEDS

8.1 PROGRAM BACKGROUND

The Common-Module VHSIC Integrated System (CVIS) was originally an Army program that was borne out of DARPA's VHSIC Program. The CVIS hardware consists of a configurable computer containing a VME-type backplane called the SC Bus, and PI-Bus backplane, and a TM-Bus. Figure 8.0 shows the block diagram of a completed unit. Since is was designed in 1988, it has been used in the Air Force ATLAS Program for a system computer and array processor.

8.2 DIGITAL TESTBED

We selected the CVIS System Processor (SP) Module for the digital testbed. The SP Module is built around a MIL-STD-1750 CPU. It also contains RAM, ROM, and I/O functions.

8.3 HYBRID TESTBED

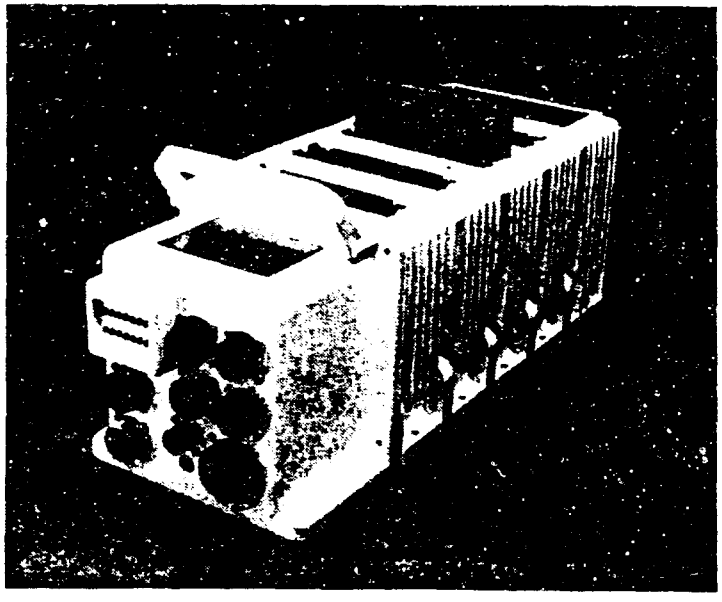
One of the CVIS functions is processing IR or visible light TV images. Video is digitized in the Video Input (VI) Module. The VI Module has two video inputs, along with the needed gain and offset functions to provide video to its high speed 8-bit analog-to digital converter. Figure 8.1 shows the VI Module functions. The VI Module contains and Inmos T800 CPU and double RAM buffer which can be used in conjunction with added circuitry to affect the overall BIT functions.

8.4 BIT TECHNIQUES APPLIED

Three BIT techniques were applied to the SP Module: EDCC, Microprocessor BIT, and Digital Wraparound BIT. EDCC was applied to the 16-bit RAM contained in memory block 1. Memory block 2 hosted the microprocessor BIT test code. Digital wraparound was applied to the discrete I/O lines in the I/O block. Three BIT techniques were applied to the VI Module: Analog Wraparound BIT, On-board ROM BIT, and Comparator BIT. Analog wraparound was added by inserting an analog switch to route the output t of the offset DAC to the ADC. On-board ROM was applied by routing the ROM output to the offset control DAC input and observing the ADC output directly. The first two techniques cannot be applied concurrently. The difference between them is that in the analog wraparound case, the T-800 CPU controls the test. The on-board ROM BIT technique uses its own logic to control the testing and to report the pass/fail. Likewise, the comparator BIT circuitry controls the test and reports the result.

8.5 IMPACT OF DESIGN MODIFICATIONS

Each testbed was used to demonstrate three BIT techniques. The BIT circuitry added to the SP Module would add an appreciable amount of circuitry and slow the processor down due to the EDCC throughput delay. Microprocessor BIT was added by merely increasing the size of the ROM to accommodate added test software. The digital wraparound components added a small amount to the overall board area. The VI Module is different because the three BIT techniques that were demonstrated cannot be added simultaneously. If BIT were to be added to the VI Module, the analog wraparound technique would be selected because it adds fewer parts.



MODULE CHARACTERISTICS

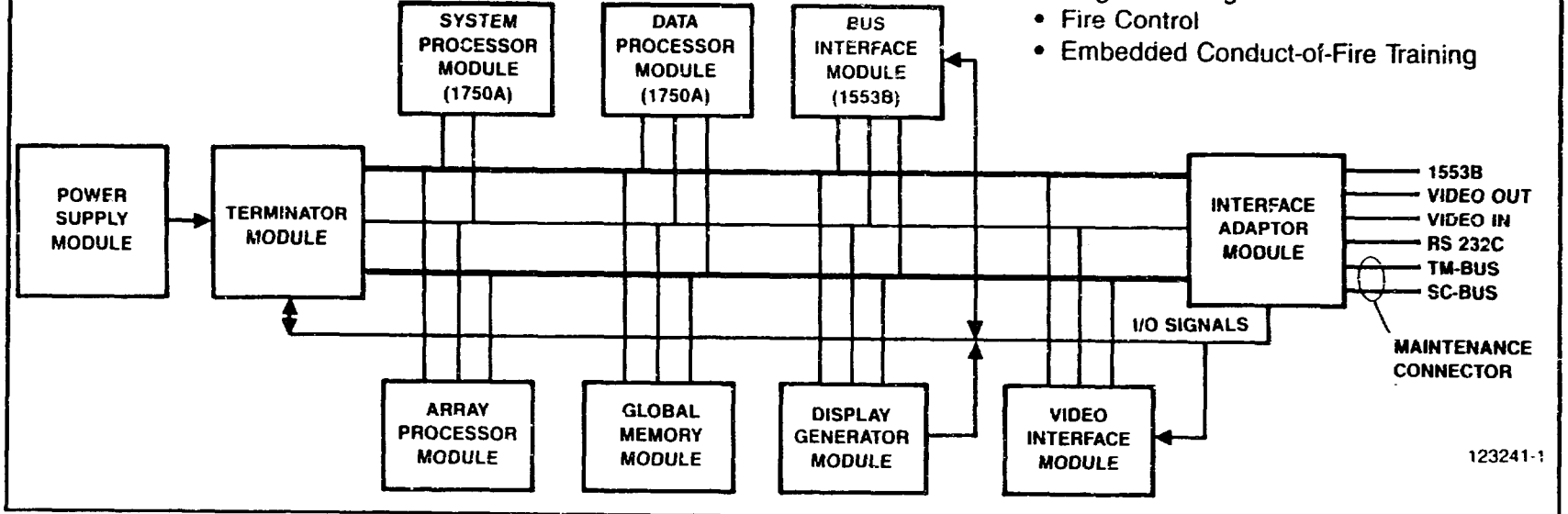
- SP: 1.9 MIPS (DAIS)
- DP: 2.5 MIPS (DAIS)
- AP: 160 MOPS
- GM: 2.5 Megabytes, Volatile & Nonvolatile
- VI: 8-BIT RES, 20 MHz, RS170
- DG: 37K Char/Sec, Graphics, RS170
- BI: BC/RT/BM

FEATURES

- VHSIC Technology/JAIWG Standards
- Plug-In Modules/Configurable
- Parallel Processing (AP module)
- Nonvolatile Data Storage
- Ada Programmable

APPLICATIONS

- Target Detection
- Target Tracking
- Fire Control
- Embedded Conduct-of-Fire Training



274_844_46

Figure 8.0 VHSIC Integrated System Block Diagram

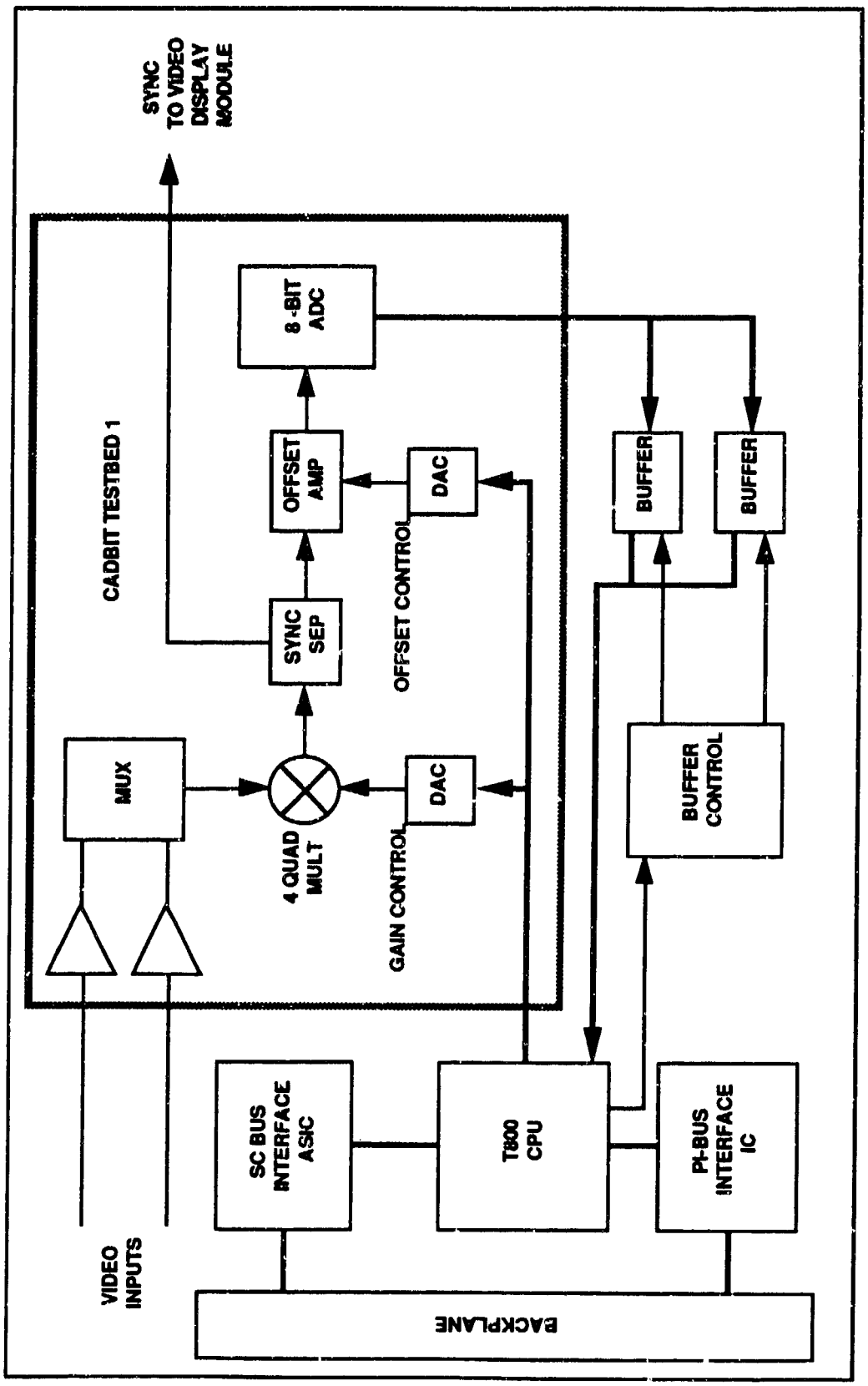


Figure 8.1 Video Input Module Block Diagram

8.6 VHDL MODEL

VHDL is an acronym which stands for VHSIC Hardware Description Language. VHSIC stands for Very High Speed Integrated Circuit. VHDL is a language for creating models of hardware designs and it contains features found in modern programming, hardware description and CAE design and simulation languages. System-1076 is a Mentor Graphics design environment for creating, analyzing and debugging models in written VHDL. An example VHDL design file has been created and stored within the System-1076 environment. This example file consists of a combination of 'inverter', 'and', 'or', and 'exclusive or' gates and functions as a data selector one out of four multiplexer with two control lines. It was demonstrated that the System-1076 as one of the functionalities of CADBIT II software environment accepted and successfully compiled this VHDL design file with no error. Since VHDL based tools are becoming widely available on platforms ranging from personal computers to multi-user Unix or Ultrix machines, this capability is essential for future testability research.

9.0 CONCLUSIONS AND RECOMMENDATIONS

The CADBIT II System provides a useful tool to automate the process of integrating BIT in Printed Circuit Board (PCB) designs. The CADBIT II System is fully integrated with Printed Circuit Board design software operating on a state-of-the-art CAD workstation. The CADBIT II System performs four major functions to aid designers in implementing BIT:

1. On-line tutorial to increase designer awareness and understanding of BIT techniques.
2. Identification and ranking of suitable BIT techniques for a given PCB design.
3. Insertion of designer-selected BIT circuitry into the PCB schematic.
4. Evaluation of the impact of added BIT circuitry in terms of increased board area, weight and power consumption.

The CADBIT II design incorporates open standards such as: Unix, the C programming language, OSF/Motif, X Windows, and VHDL. The use of these open standards ensures the developed CADBIT II System is transportable to other CAD workstations and PCB design software. The CADBIT II System incorporates a BIT library utility so that changes, modifications or additions to the BIT library can be performed on-line. The power and flexibility of the CADBIT II System is derived from the data in this BIT Library. The CADBIT II System provides the baseline for Rome Laboratory to expand the BIT Library and integrate the CADBIT II System with other CAD/CAE tools.

During the CADBIT II contract, HMSC funded and submitted a White Paper summarizing results of a user survey for proposed enhancements to the CADBIT II System. Tables 9.0 and 9.1 divide these enhancements into near-term and long-term growth. Funding should be procured to perform these enhancements as well as integrate the CADBIT II System with other RL tool kits

TABLE 9.0 NEAR-TERM GROWTH AREAS

Include additional parameters in BIT selection and evaluation functions
Provide the designer with tested "canned" BIT routines to aid BIT implementation
Provide designers additional BIT data related to hardware components
Provide additional data to the designer during the insertion process
Highlight design violations where pins or nodes cannot be tested

TABLE 9.1 LONG-TERM GROWTH AREAS

Provide design rule checks to aid designer in BIT evaluation process
Handle sharing of BIT circuitry among modules
Integrate system testability requirements and analysis tools
Reduce the number of errors made by the designer during the BIT insertion process
Incorporate improved HDLs such as VHDL
Provide reliability data for BIT inserted on the CUT to evaluate the impact of the BIT circuitry on reliability

10.0 REFERENCES

Document No.	Document Title	Date
HMSC-SDP-91-003	CADBIT II Software Development Plan	16 Oct 92
HMSC-SDD-91-003	CADBIT II Software Design Document	15 Oct 92
HMSC-STP-91-005	CADBIT II Software Test Plan	8 Oct 92
HMSC-SUM-91-003	CADBIT II Software User's Manual	9 Oct 92
RADC-TR-89-209 Vol 2	CADBIT I BIT Library Package	Oct 89
RADC-TR-89-209 Vol 3	CADBIT I Software Requirements Specification	Oct 89

MISSION

OF

ROME LABORATORY

Rome Laboratory plans and executes an interdisciplinary program in research, development, test, and technology transition in support of Air Force Command, Control, Communications and Intelligence (C^3I) activities for all Air Force platforms. It also executes selected acquisition programs in several areas of expertise. Technical and engineering support within areas of competence is provided to ESD Program Offices (POs) and other ESD elements to perform effective acquisition of C^3I systems. In addition, Rome Laboratory's technology supports other AFSC Product Divisions, the Air Force user community, and other DOD and non-DOD agencies. Rome Laboratory maintains technical competence and research programs in areas including, but not limited to, communications, command and control, battle management, intelligence information processing, computational sciences and software producibility, wide area surveillance/sensors, signal processing, solid state sciences, photonics, electromagnetic technology, superconductivity, and electronic reliability/maintainability and testability.